THESIS

MOBILE SITUATIONAL AWARENESS TOOL: UNATTENDED GROUND SENSOR-BASED REMOTE SURVEILLANCE SYSTEM

by

Bradley C. Palm
Ryan P. Richter

September 2014

Thesis Advisor:	Gurminder Singh
Co-Advisor:	John Gibson

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2014	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE MOBILE SITUATIONAL AWARENESS TOOL: UNATTENDED GROUND SENSOR-BASED REMOTE SURVEILLANCE SYSTEM	**5. FUNDING NUMBERS**
6. AUTHOR(S) Bradley C. Palm, Ryan P. Richter	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The listening post/observation post is a method employed by infantry units in combat to increase battlefield situational awareness and prevent surprise by the enemy. This technique is costly to the employing unit in terms of manpower requirements and increased risk to friendly personnel. To reduce these costs, we created a prototype, the Mobile Situational Awareness Tool, that combined commercial off-the-shelf components with wireless unattended ground sensors for the purpose of automating the listening post/observation post for the tactical infantry unit.

The prototype system incorporated wireless sensor node prototypes created by the Defense Advanced Research Projects Agency, originally intended for the creation of a smart minefield. A web application was created using a custom Node.js server that enabled cross-platform monitoring of the system by warfighters in the field with mobile smart-devices to include smart-phones and tablets.

Field-testing of the prototype showed the system capable of detecting and classifying intruders in the sensor field but revealed that more robust threat classification algorithms utilizing multiple sensor modalities would yield a greater degree of automation and autonomy.

14. SUBJECT TERMS Unattended Ground Sensors, Wireless Sensor Network, Mesh Network, Ad-Hoc Network, Automated Listening Post/ Observation Post, Intelligence, Surveillance, and Reconnaissance, Single Page Application, Web Application, Responsive Web Design, Mobile Monitoring Platform	**15. NUMBER OF PAGES** 141
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

MOBILE SITUATIONAL AWARENESS TOOL: UNATTENDED GROUND SENSOR-BASED REMOTE SURVEILLANCE SYSTEM

Bradley C. Palm
Captain, United States Marine Corps
B.S., University of St. Thomas, 2008

Ryan P. Richter
Captain, United States Marine Corps
B.B.A., University of San Diego, 2008

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2014

Authors:	Bradley C. Palm Ryan P. Richter
Approved by:	Gurminder Singh Thesis Advisor
	John Gibson Co-Advisor
	Peter Denning Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The listening post/observation post is a method employed by infantry units in combat to increase battlefield situational awareness and prevent surprise by the enemy. This technique is costly to the employing unit in terms of manpower requirements and increased risk to friendly personnel. To reduce these costs, we created a prototype, the Mobile Situational Awareness Tool, that combined commercial off-the-shelf components with wireless unattended ground sensors for the purpose of automating the listening post/observation post for the tactical infantry unit.

The prototype system incorporated wireless sensor node prototypes created by the Defense Advanced Research Projects Agency, originally intended for the creation of a smart minefield. A web application was created using a custom Node.js server that enabled cross-platform monitoring of the system by warfighters in the field with mobile smart-devices to include smart-phones and tablets.

Field-testing of the prototype showed the system capable of detecting and classifying intruders in the sensor field but revealed that more robust threat classification algorithms utilizing multiple sensor modalities would yield a greater degree of automation and autonomy.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

ADAPT	adaptable sensor system
AJAX	asynchronous JavaScript and XML
AP	access point
BYOD	bring your own device
C2	command and control
CBP	U.S. Customs and Border Protection
CFF	call for fire
C-IED	counter-improvised explosive device
COC	combat operations center
COTS	commercial off-the-shelf
CSS3	cascading style sheets version 3
DARPA	defense advanced research projects agency
DHS	Department of Homeland Security
DOM	document object model
ECBC	Edgewood Chemical Biological Center
FDMA	frequency division multiple access
GCSS	global combat support system
GPS	global positioning system
HMMWV	high mobility multipurpose-wheeled vehicle
HTML5	HyperText Markup Language 5
ISR	intelligence, surveillance, and reconnaissance
IV	inter-visual
LAMP	Linux, Apache, MySQL, PHP
LP/OP	listening post / observation post
MAC	medium access control
MAGTF	Marine air-ground task force
MCRP	Marine Corps reference publication
MCWP	Marine Corps warfighting publication
MEF	Marine expeditionary force
MJPEG	motion joint photographic experts group
MOOTW	military operations other than war

MSAT	mobile situational awareness tool
MSS	mobile surveillance system
NAT	network address translation
OIG	Office of Inspector General
ORP	operational rally point
OTA	over the air
OTIA	Office of Technology Innovation and Acquisition
PB	patrol base
PIR	passive infrared sensor
PSK	pre-shared key
RFI	request for information
RVSS	remote video surveillance system
RWD	responsive web design
SBC	single board computer
SCAMP	sensor control and management platoon
SIS	shared information space
SMMS	sensor mobile monitoring system
SPA	single page application
T&R	training and readiness
TAOR	tactical area of operations
TCP/IP	Transmission Control Protocol/Internet Protocol
TDMA	time division multiple access
TLS	transport layer security
TO/TE	task organization and table of equipment
TRSS	tactical remote sensor system
TTP	tactics, techniques, and procedures
UDP	user datagram protocol
UI	user interface
UGS	unattended ground sensor
USBP	United States Border Patrol
WSN	wireless sensor network
XML	extensible markup language

ACKNOWLEDGMENTS

We would like to thank our wives and families for their unending support.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Situational awareness is essential to the warfighter to achieve victory on the battlefield. Combat leaders require situational awareness to make timely and accurate decisions. Faster and better decisions lead to momentum and the outpacing and destruction of the enemy. One of the most fundamental ways that a ground warrior increases his situational awareness is through listening and observation posts (LP/OPs). LP/OPs are small groups of troops (usually 2–4 men) that are emplaced in an area to observe a particular sector of the battlefield that is not observable by the main body of a unit. This extends the range of observation of the unit and allows for earlier detection of enemy activity, thus improving the awareness of the unit leader. The utilization of LP/OPs is a basic and proven combat tactic. Requiring no specialized training or equipment, conducting LP/OPs is a capability inherent to any infantry unit. American combat forces have utilized them throughout history.

LP/OPs are utilized in a wide variety of military operations. In defensive operations, they are emplaced forward of defensive lines in order to see beyond intervening terrain and give advanced warning of approaching attackers. Used in offensive operations, they are emplaced on the flanks of an ambush position in order to report the direction of the enemy's approach. They are also frequently used in raids, vehicle checkpoints, strongpoints, and counter-improvised explosive device (C-IED) operations.

LP/OPs have disadvantages. They require that troops be separated from the main body of the unit, taking away combat power that can be massed against the enemy. Though more LP/OPs expand the range of observation for the unit leader, each LP/OP that is employed takes troops away from the main engagement area. Thus, manpower limitations place an upper bound on the total number of LP/OPs that can be emplaced by a single unit. Also, isolating the troops of the LP/OP from their unit makes them more vulnerable to being killed, wounded, or captured [1]. The isolation of the LP/OP means that it is less capable of receiving adequate supporting fires from the employing unit in the event of enemy contact or engagement. Also, having friendly forces forward of enemy lines occupying these LP/OPs can constrain (mask) the fires of the friendly unit and complicate geometries of fire. This means that because friendly LP/OPs are located between friendly forces and the enemy, LP/OPs run a significant risk of fratricide if friendly forces do not carefully coordinate their fires. Finally, because LP/OPs must be continually manned, they can interfere with the sleeping patterns of troops by altering the rest-plan of a unit over long periods of operation. The stress of maintaining LP/OPs permeates all levels of leadership within a unit. LP/OPs require significant time spent planning and coordinating at the upper-leadership levels of a small, tactical unit. At the lowest levels, being isolated from the main body of the unit and the disruption of regular sleep cycles can have a negative effect on morale.

Automation through sensors and computing technology could augment or potentially replace the manned LP/OP. A surveillance system that is compatible with a variety of commercial off-the-shelf (COTS) smart devices could simultaneously decrease manpower requirements and reduce risk to friendly personnel who would otherwise need to man the isolated LP/OPs. This system could provide enhanced situational awareness for combat leaders through a real-time monitoring capability.

With much research underway regarding smartphone use on the battlefield [2], this system would leverage the anticipated proliferation of personal handheld smart devices on the on the battlefield and would mitigate the requirement for specialized monitoring equipment characteristic of other remote surveillance systems. Since monitoring of such a system would be conducted on general-purpose smart devices, this system would reduce the amount of gear that has to be carried, powered, and maintained by the warfighter operating in an austere environment. Furthermore, all members of a unit possessing a smart device and within range of the network could access the system's monitoring interface simultaneously, enabling maximum situational awareness within the unit. This is not possible for a sensor system with only a limited number of specialized and dedicated monitoring devices.

Previously developed technology related to ground sensors does not adequately address the need for automation of the LP/OP at the small unit level with the ability for distributed, cross-platform monitoring through the use of COTS smart devices. Consisting of a suite of seismic and infrared intrusion detecting sensors, relay devices, and monitoring equipment, the Tactical Remote Sensor System (TRSS) is the primary sensor system employed by the Marine Corps [3]. However, TRSS was not designed to be utilized by small units to increase local situational awareness and influence tactical battlefield decisions. Rather, the TRSS is a high-level intelligence collecting asset, meant to be employed at the Marine Air-Ground Task Force (MAGTF) level and tightly controlled by sensor control and management platoons (SCAMP) [3]. Additionally, the TRSS is not interoperable with COTS smart devices and requires specially built monitoring equipment. The two types of monitoring tools for TRSS are the sensor mobile monitoring system (SMMS)—consisting of a high mobility multipurpose wheeled vehicle (HMMWV) with a full load of dedicated communications equipment—and the portable monitor, which is a handheld display unit designed primarily for testing the sensors during emplacement. The need for dedicated monitoring equipment unnecessarily adds to the combat load of dismounted infantry and to fuel requirements when the SMMS is employed: the SMMS weighs 7,785 pounds, while the portable monitor weighs five pounds per unit [3].

The TRSS also requires extensive training that makes it impractical for use by basic infantry riflemen. TRSS operators in the Marine Corps are currently required to attend five weeks of training at the Remote Sensor Operations Course. This level of required training means that only SCAMP platoons consisting of specially trained sensor operators are authorized to employ these systems. However, the shortage of SCAMP platoons means that the majority of infantry companies in the Marine Corps will not have access to the personnel required to operate these sensors. Even with sufficient sensor operators, the Marine Corps allocates only 600–800 total sensors per Marine expeditionary force (MEF), which equates to roughly 50–70 sensors per infantry battalion, or 16–23 sensors per rifle company. One reason for this shortage is that TRSS sensors are expensive ($1,020,847.30 per unit) [4], even though [3] describes them as expendable. This cost keeps sensors out of the authorized inventory of infantrymen.

The United States Border Patrol (USBP) is another organization that has looked to leveraging technology in order to detect intrusion across border areas by unauthorized personnel, a task that is similar to conducting a military LP/OP [5] to detect enemy activity. The USBP employs a combination of technologies to monitor the southern border of the United States, including the remote video surveillance system (RVSS), the mobile surveillance system (MSS) and unattended ground sensors (UGS) [6]. The RVSS is a system of fixed camera positions with the ability for remote monitoring and pan-tilt control of the cameras, while the MSS consists of a camera mounted on a pole fixed to a flatbed truck. Similar to the TRSS, all of these systems are relatively expensive, not fully dismountable by foot soldiers, and not able to be monitored by a multitude of commercial handheld devices.

B. OBJECTIVE

This thesis provides a reference design for an automated LP/OP to show the viability of a portable UGS system for employment by the basic infantry rifleman that automates or otherwise enhances the LP/OP to increase the situational awareness of the Marine rifle squad. The mobile situational awareness tool (MSAT) is a prototype system consisting of networked sensors, surveillance cameras, and a portable application server with monitoring software that is compatible with a variety of COTS smart devices. This system is inexpensive, modular, mobile, and user-friendly. It allows monitoring and surveillance of the battlefield by combat warriors without the need for specialized training, and has added functionality such as the tracking of friendly forces and tactical chat capabilities, with the ultimate goal of maximizing situational awareness on the ground.

MSAT incorporates a network of wireless sensor nodes that are capable of automatic network formation and are enabled with passive infrared sensors (PIR) for its primary means of intrusion detection. The Defense Advanced Research Projects Agency (DARPA) developed these sensor nodes. After intrusion detection, threats are classified through the use of video cameras that are integrated into MSAT, by the operator. The system uses an application server to process data from the sensor nodes and allows the operator to monitor the sensor field from a handheld device. The whole system is light enough to be transported completely by dismounted Marines. Furthermore, MSAT is intended to be compatible with the greatest number of commercial smart-devices, to include tablets and phones running Android and iOS operating systems.

MSAT was built for the particular use case of a Marine rifle squad conducting a defensive battle position, a core task defined by the Marine Corps training and readiness manual (T&R) [7]. In such a scenario, the rifle squad's mission would be to repel an enemy assault by fire and close-combat. Though intelligence would provide the squad with a general idea as to the size and disposition of the enemy, as well as a general estimate of the location of this enemy, the nature of defensive operations is that the exact time of arrival, direction of approach, and makeup of the enemy would be uncertain. For this purpose, the squad leader must emplace LP/OPs apart from his main battle positions in order to provide early warning of an enemy approach or to warn of enemy infiltrating from the rear. These LP/OPs also provide the squad leader with the ability to see into dead space, or ground that is not observable due to masking by intervening terrain. The farther out the squad leader places his LP/OPs, the sooner he is able to become aware of the enemy and make maneuver decisions in order to most efficiently and effectively engage the enemy. However, this carries an increased risk to the Marines of the LP/OP due to the declining ability of the main battle position to support the LP/OP with fire in the event of enemy contact. Also, because the rifle squad consists of only 13 Marines, and because no position can be manned alone according to the Marine Corps buddy-team philosophy, the squad leader can realistically emplace at most two LP/OPs.

In the defensive scenario, employing MSAT would enhance the security and situational awareness of the squad without excessive manpower requirements. Ground sensors and cameras can be emplaced farther out from the defensive position without concern for Marines being isolated. Also, without Marines forward of the battle position, there would be less risk of fratricide and fewer restrictions on fires. The squad can emplace sensors to its rear, its flanks, in surrounding dead space (e.g., draws, holes) and over intervening terrain to detect enemy in areas that would otherwise be unobservable (i.e., areas identified in the modified combined obstacle overlay). Since the squad is not restricted to two or fewer LP/OPs, many more unobservable areas can be monitored through the use of the system. Each of the squad’s three fire team leaders would be able to monitor the sensor emplacements in real-time from his own handheld device and observe evidence of enemy activity as it occurs. In this way, the distribution of situational awareness information is automated, reducing voice radio traffic and confusion. It also reduces the potential for noise required for verbal communication (either direct or via radio), which can risk loss of the friendly element of surprise and stealth. MSAT would function in low-visibility situations (e.g., nighttime, poor weather) due to the versatility of the sensors.

As a supplement to our use case, the MSAT would have the ability to be monitored from a remote combat operations center (COC). This means that military personnel would be able to monitor the system from outside the local area of operations. This would allow higher-level commanders to access real-time data on battlefield events, thus increasing overall situational awareness in support of their ability to make better decisions, anticipating the needs of the units about to encounter or engage enemy combat elements. This capability, though, presents a networking challenge due to the need to establish connectivity outside of MSAT while operating in a potentially infrastructure-less environment. Thus, a mobile ad hoc networking capability may be essential.

Currently, there is no system being fielded by the Marine Corps with the ability to execute the presented use case. Since the TRSS is unavailable to rifle squads, is too expensive and complex to operate without specialized operators, and cannot be monitored by multiple Marines simultaneously through the use of tactical smartphones, it is inappropriate to our use case.

MSAT has several potential advantages. The relatively low cost of the system due to the utilization of existing COTS components would make widespread adoption by small infantry units feasible. The autonomous nature of the UGS-component makes sensor field setup comparatively less risk-prone than sensor options requiring more complex installation emplacement and setup. The ability to monitor the system from a wide variety of devices and from multiple devices simultaneously is novel and enhances the potential for increased situational awareness by the squad, while at the same time reducing the combat load for the infantry squad. MSAT also leverages the familiarization most modern junior Marines have with smartphones and tablets to design an interface that is intuitive and requires little specialized training [8]. Lastly, the utilization of an asynchronous, non-blocking application server and WebSockets provides operators with a fast and accurate monitoring application.

Testing the system highlights multiple weaknesses or areas for improvement for MSAT. The system requires constant electrical power from batteries in order to power the sensors' GPS and radios, and power is also required to provide a constant Wi-Fi bubble to allow remote communication access to the sensor field. Thus, operating time is likely too short in its current state for actual deployment of the system in combat and power saving strategies will have to be implemented.

C. THESIS ORGANIZATION

The rest of this thesis explores in detail the problem of developing a usable surveillance tool for the modern foot soldier and presents MSAT as a possible solution. Chapter II discusses how other UGS system implementations in use today fail to adequately solve the problem of automating LP/OPs for infantrymen. Technologies and related architectures that utilized in building the MSAT prototype are then presented.

Chapter III analyzes the task of conducting the LP/OP, as defined by the Marine Corps Infantry T&R manual and broadly defines the specification and testing requirements for the MSAT reference design.

Chapter IV describes the design, construction, and testing of the MSAT system. Different iterations of the system are discussed, along with rationale for design choices and considerations, to include the successes and failures encountered during the developmental phase. Chapter IV also presents the performance of MSAT through testing. Finally, it discusses potential weaknesses and shortfalls of the system.

Chapter V, the conclusion, provides summary remarks regarding MSAT and suggests future work to improve the system.

II. BACKGROUND INFORMATION

A. PROBLEM DOMAIN

1. History

An inherent action in warfare is to make threat assessments and attempt to mitigate the identified risks. Arguably, the use of LP/OPs can be traced back to the beginnings of warfare. A history book of the Roman Empire describes an early observation post, recounting, "The ground floors of these towers were used as living quarters by the garrison of auxiliary soldiers (cavalry and infantry), while the upper floor, furnished with a gallery, served as an observation-post from which the enemy could be watched and signals given by means of torches" [9]. Another example of an early LP/OP comes from the American Revolution, when colonial insurgents utilized observation posts on Long Island to warn of inbound British Troops [10]. During the challenging jungle fighting of Vietnam, LP/OPs were used extensively to facilitate early warning in the dense vegetation that surrounded patrols and outposts [1].

The current employment of LP/OPs remains relatively unchanged from the early days of warfare. LP/OPs in Afghanistan have been utilized to detect and disrupt the enemy. They were reinforced with powerful optics that increased the range of observation and allowed troops to view the surrounding terrain, even in low-light conditions. Commanders in charge of combat outposts (COPs) utilized LP/OPs to deny insurgent observers key terrain from which to coordinate indirect fire, and also prevent the occupation and utilization of machine gun positions. Upon the unit leader determining the necessity of LP/OPs as part of the defensive plan, leaders had to determine the best way to support these positions with fires and logistics. Due to the ruggedness of the Afghan terrain, which had the tendency to isolate American troops due to strained communications [11], this proved a difficult challenge. Considerations included the rotation schedule required to staff the position or positions, the amount of logistical materials needed to keep the position combat effective, and the positioning of the unit's weapon systems and coordination of geometries of fire. After the initial planning and establishment of the LP/OPs, adjustments inevitably needed to be made to reinforce the overall defensive plan.

2. LP/OP Tactics, Techniques, and Procedures

The *Infantry Training and Readiness (T&R) Manual* (NAVMC 3500.44A) outlines the individual and collective training requirements that Marine Corps infantry units will train toward in preparation for combat. One of the tasks found in the T&R manual, titled INF-MAN-3102, is to conduct a listening post/ observation post. The T&R manual outlines the condition, the standard, and the event components. The condition for executing the LP/OP is stated as, "Given a unit, an order, and supporting a defensive scheme of maneuver during daylight and limited visibility" [7]. The standard is "to provide early warning while seeking to avoid direct enemy contact" [7]. The event components are as follows: "conduct planning, conduct resupply, prepare for combat, execute command and control, conduct a passage of lines, move to the LP/OP, conduct link up as required, conduct relief in place as required, occupy the LP/OP, establish security, conduct weaponeering, deconflict battlespace geometry, maintain communications, improve positions as necessary, confirm prescribed routes to friendly lines, provide early warning, report information, break contact as required, move along prescribed route(s) back to defense, conduct passage of lines, conduct post combat actions" [7]. When all these tasks are capably performed and validated by a commanding officer, a Marine unit is considered proficient in the execution of the LP/OP.

A similar Army publication [12] delves deeper into describing the conduct of the LP/OP, which is valuable in understanding the implementation of such a position. According to [12], the first task is to properly select the location of the LP/OP. The location should provide maximum observation of the assigned battle space. The location should provide cover, concealment, and protection for the troops occupying the position. It is also important that the route to and from the LP/OP be covered and concealed, so as to allow troops to safely occupy, rotate into and out of, provide resupply, and egress from the position. The location also needs to be within range of supporting fires by the main body element utilizing their direct fire weapon systems. Next, a unit leader must assign a sector of observation in harmony with the overall plan, determine what kinds of things the troops should be observing, and detail what needs to be reported to higher command. The LP/OP should be robust enough to operate during low light conditions. It is important, when selecting a position, to avoid obvious terrain that would draw attention to the presence of the LP/OP and alert the enemy. An in-depth communication plan is needed for the LP/OP, which includes redundant methods of communication. Wireless radio communication, pyrotechnics or smoke to create a signal, and messengers can all be used as a means of reporting.

Once these considerations are satisfied, the LP/OP plan is executed. At a minimum, it is manned by two troops. For larger observation areas or longer periods between relief, a fire team of four troops may be necessary to man the position. While in the position, one troop is responsible for scanning the terrain and looking for anomalies while their partner records the information. The troop acting as the observer should be rotated every 20–30 minutes to reduce fatigue and increase attention. The LP/OP team should be relieved every 2–4 hours, keeping in mind that each tactical scenario will drive this determination. These details on the execution of the LP/OP highlight a very deliberate planning process and taxing execution cycle that ultimately takes a toll on those conducting the operation. Due to the inherent complications of utilizing human beings in LP/OPs, there has been an attempt to apply technology toward this necessary combat task in an overall effort to reduce the numerous pitfalls of requiring troops to man these positions.

3. Introduction of Technology

The use of ground sensors in military applications dates back to the Vietnam era and became public knowledge on April 17, 1967 [13], in a Newsweek article, and was again alluded to on September 7, 1967 [13], when Secretary of Defense Robert McNamara gave a press conference vaguely describing the concept of an electronic barrier that was going to protect South Vietnam. This barrier was to be emplaced along the demilitarized zone and the Ho Chi Minh trail to monitor enemy troop traffic, since the carpet bombing raids of North Vietnam were ineffective against thwarting the enemies' ability to mobilize troops along these routes [14]. The creation of this barrier was envisioned by a think tank organized in 1959, named the Jasons, which proposed a hybrid approach of using ground positions, pinpointed bombing raids, air-laid mines, and the use of battery-powered sensors [14]. This "highly theoretical" plan (e.g., code name Igloo White), as stated by Military Assistance Command Vietnam Commander General Westmoreland, was opposed by the Navy, Air Force, and the Marines [14]. The Marines especially opposed the project, since the barrier's location was in their battle space, and they were responsible for providing troops and resources to the construction efforts. However, sentiments quickly changed in regard to the new technology when the Marines were provided the sensor systems in an effort to defend their outpost during the Battle of Khe Sanh. After a month of flight operations, 316 acoustic and seismic sensors were air delivered, equating to a total of 44 sensor strings [14]. The Marines were impressed with the sensors' ability to accurately track the enemy and credited the sensors with providing forty percent of the actionable intelligence to the fire support coordination center [15]. They also credited the sensors with preventing another fifty percent loss of Marine forces [15]. On one such occasion, sensors indicated that a massive troop formation was assembling near a remote hilltop outpost. Artillery used this information to prevent the enemy assault and inflict heavy losses on the North Vietnamese soldiers. The commander of the 26th Marine Regiment, Colonel David Lownds, recounted this event at Khe Sanh stating:

> The sensors which had been emplaced on Route 9 to the Laotian border suddenly came to life and it became obvious that a large column was moving adjacent to Route 9 toward the base… By computing the length of the column by information produced by the sensors, it became obvious to me that an enemy regiment was trying to close the base. This information coupled with possible assembly areas, allowed us to bring down upon this unit devastating firepower to breakup the impending attack. [16]

This was arguably the Marine Corps', as well as any of the services', first exposure to automated sensor systems in an early warning role and its effectiveness was proven through some of the most intense combat that American forces faced during the Vietnam War. Igloo White operations on the trail continued after this, in support of the

Air Force strategic bombing campaign, that targeted troop movements and logistical resupply convoys along the Ho Chi Minh trail [14]. Overall, there were many localized success stories involving the use of sensor systems in Vietnam; however, the price tag of over one billion U.S. dollars per year and the minimal effect on the enemy at the strategic level lead members of Congress and the military to question its future use [17].

After Vietnam, there were stock piles of sensors and supporting communications equipment, and when a situation presented itself where the devices could be employed for military operations other than war (MOOTW), America was eager to test them in a peace keeping role [18]. When Israel and Egypt agreed to negotiations with the U.S. Secretary of State, and a peace treaty was signed, there was a buffer zone created and the use of ground sensors was utilized to enforce the treaty that both parties had signed [18].

Following the Sinai Treaty Agreement, there were no recognized uses of ground sensors in combat scenarios or MOOTW, until Operation Iraqi Freedom (OIF) in Iraq and Operation Enduring Freedom in Afghanistan (OEF). The border patrol had started surveillance of America’s borders with sensor systems in the mid-1970s, and it continues to do so today [18]. The gap in sensor usage between the Vietnam/Sinai eras and the Iraq and Afghanistan wars can be attributed to the gradual improvement of sensor, communication, and computer technology that made sensors a more viable possibility in the modern era. Sensor systems were steadily researched and developed until reaching a culminating point when devices were made small enough for practical usage and with increased processing power. They also became cost efficient to the point that they could begin to be considered for actual combat use without the cost impact of losing sensors. This climate of reduced size and cost, coupled with increased computing power, ushered in an advanced age where sensors began to realize high demand across current battlefields.

B. SIMILAR ATTEMPTS TO SOLVE THE PROBLEM

1. Tactical Remote Sensor System

The military recognized the importance of the wireless sensor network and unattended ground sensor technology breakthroughs and sought to integrate them into current operations. Since the Marine Corps' first introduction to UGS in Vietnam, it more recently fielded the TRSS, which is employed by the Ground Sensor Platoon (GSP). This system was composed of a suite of sensors that offered detection of human activity and movement of vehicles in real time. Due to it being a Marine Corps surveillance asset, the actual specifications and limitations of the system were not advertised, but it is public knowledge that the sensors have the ability to monitor an environment for at least thirty days [4]. The system, as dictated by the task organization and table of equipment document (TO/TE), was a MEF intelligence asset and therefore very difficult to utilize at the small unit tactical level given that there were only 6 TRSS systems allocated per MEF. According to the Marine Corps Global Combat Support System (GCSS), one system cost $1,020,847.30. Equipment in a sensor set consisted of both a vehicle monitoring system (A2306) and unattended ground sensors (A3255); both are shown below in Figure 1. Each TRSS unit consists of 24 Seismic Intrusion Detectors (SID), 24 Infrared Intrusion Detectors (IRID), 24 Air-Delivered Seismic Intrusion Detectors (ADSID), four portable monitors, and one Sensor Mobile Monitoring System (SMMS) [4].

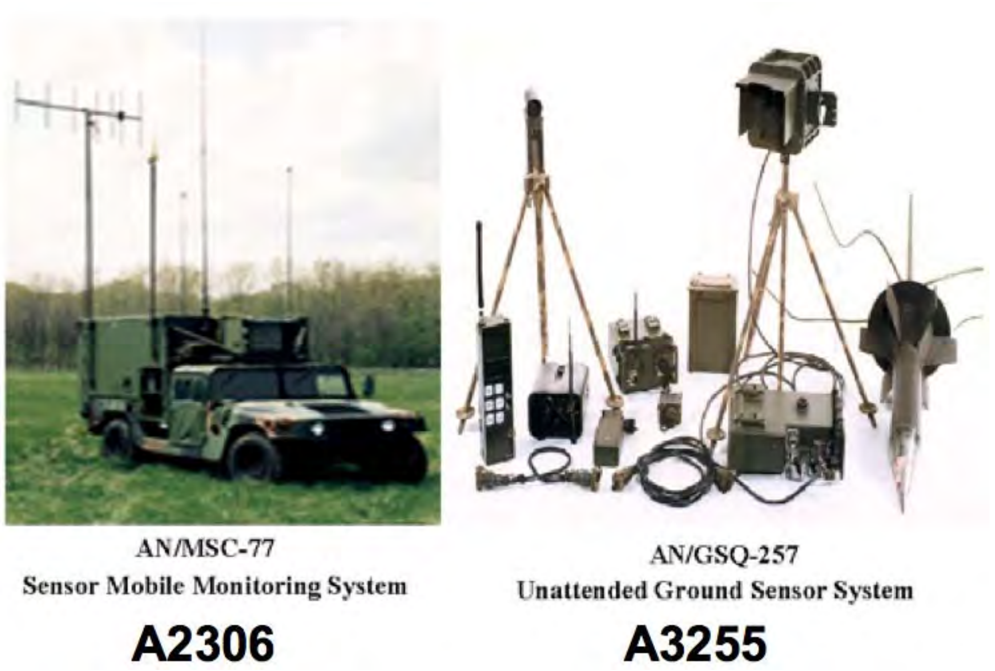

AN/MSC-77
Sensor Mobile Monitoring System
A2306
AN/GSQ-257
Unattended Ground Sensor System
A3255

Figure 1. USMC TRSS Components, from [4]

Unfortunately, the mobile monitoring system was mounted in an HMMWV, which did not lend itself well to transport within the confined areas where dismounted troops were likely to operate. The system, compared to current sensor technology, can be considered outdated, costly, bulky, and training intensive—it required members of the GSP to be trained for a period of thirty-five days to properly employ the devices [4], a period of time that would make widespread training of personnel within Marine infantry battalions impractical.

2. Border Patrol

The USBP was another organization that invested heavily in technology capable of detecting intrusion across border areas by unauthorized personnel. The USBP employed a combination of technologies to monitor the southern border of the United States, including RVSS, the MSS and UGS [6]. The RVSS was a system of fixed camera positions with the ability for remote monitoring and pan-tilt control of the cameras, while the MSS consisted of a camera mounted on a pole fixed to a flatbed truck. Similar to the TRSS, all of these systems were relatively expensive, not fully dismountable by foot soldiers, and not able to be monitored utilizing a multitude of commercial handheld devices.

On April 5, 2011, U.S. Customs and Border Protection (CBP) issued a request for information (RFI) to determine the current state of the art in UGS systems, what advances had been made, and what would be the most effective employment of the system. The CBP conveyed the importance of UGS to their mission by stating, "An integral component to this situational awareness is the use of unattended ground sensor technology that provides information on the location of potential illegal border entries." [19]. After almost two years of research and input from the market leaders in UGS technology, the Office of Technology Innovation and Acquisition (OTIA) for the CBP released a cancellation of the RFI stating, "OTIA is not planning to release a solicitation for this specific requirement in the near future" [19]. On February 11, 2013, *Wired* (a technology-oriented magazine) did an interview with CBP following this abrupt change in direction for their UGS program. CBP's public affairs officer, Jenny Burke, described the cancellation as a result of failure to resolve overlapping issues in integrating new UGS with an existing system, oversaturation of radio frequencies, and limited bandwidth. [20]. It was estimated at the time of the article that the sensor system issues highlighted would be resolved within the next six to nine months, and the CBP would again try upgrade the aging sensor system. The UGS upgrade was a necessary priority for the CBP's layered approach to defending the U.S. border, especially when considering the December 2005 Department of Homeland Security (DHS) Office of Inspector General's (OIG) report [21]. In this report, the OIG found that only four percent of the detections triggered by the sensors were illegal border crossings. During the review of a sample of the system data, it was determined that thirty-four percent of the signals were false alarms and the other sixty-two percent were unknown, due to the lack of a capability for the sensors to classify the intrusion and the unavailability of personnel to respond.

This look at the CBP sensor system highlights multiple key reasons for continued exploration in the wireless sensor networks (WSN) field. Foremost, the need for commercially available low cost sensor systems is a common requirement for projects within military and civilian entities. The technology, software, and hardware are at a state of advancement where the introduction of a highly effective WSN is feasible. Secondly, the complexity of issues surrounding the WSN domain is challenging and the fundamental problems of routing, integration, networking, and data propagation have not been solved.

C. RELATED TECHNOLOGICAL APPLICATIONS

1. Wireless Sensor Networks

Wireless sensor networks are increasing in popularity due to their ability to greatly enhance our situational awareness of our surroundings and the fact that technology has developed to a point where these devices are becoming financially practical to emplace throughout the environment. This has enticing potential within the military domain, since situational awareness is critical to maintaining surprise in offensive actions and early detection in support of defensive operations. The authors of *Wireless Sensor Networks* best sum up the challenges currently facing WSNs by saying:

> The design, implementation, and operation of a sensor network requires the confluence of many disciplines, including signal processing, networking and protocols, embedded systems, information management, and distributed algorithms. Such networks are often deployed in resource constrained environments, for instance with battery operated nodes running untethered. These constraints dictate that sensor network problems are best approached in a holistic manner, by jointly considering the physical, networking, and application layers and making major design trade-offs across the layers. Consequently, for an emerging field such as sensor networks that involves a variety of different technologies, a student or practitioner often has to be versed in several disparate research areas before he or she can start to make contributions. [22]

Another interesting relevant challenge relates to how the military will adapt and incorporate this kind of device into current military operations. The question of how the basic Marine riflemen can employ this tool is an interesting one to military officers. That question also touches upon the fundamentals of usability design within the study of Computer Science. Understanding how an end-user interacts with a WSN should be the focal point in integrating these devices successfully into combat operations. A more user-friendly approach, coupled with a lightweight employment of the device would greatly enhance the adaptation and employment of this system in tactical operations at the small unit level.

2. Defense Advanced Research Projects

Defense Advanced Research Projects (DARPA) is experimenting with Wireless Sensor Networks, particularly as part of the Adaptable Sensor System (ADAPT) program. The focus of this program is to streamline the development and procurement of intelligence, surveillance, and reconnaissance (ISR) sensor systems, within the military, so that the sensor systems can keep pace with the commercial market and allow for the best tools to be fielded for the mission. It is estimated that the typical military sensor system takes between three to eight years of development and testing before it is fielded to operational units [23]. In comparison, commercial sensor systems that are developed in a more competitive environment are fielded in one to two years. The primary goal of the ADAPT program is "To deliver common hardware and software that can be quickly configured to perform a variety of mission-specific ISR applications" [23]. The ADAPT program achieves this by focusing on a reusable hardware core, reusable software, and sensor-specific applications. The reusable hardware core is of interest because it utilizes already-available low-cost commercial components and is capable of being upgraded as the state-of-the-art advances. The reusable software addresses issues of processing, storage, navigation, communication, and orientation, which are common to a wide array of sensor systems and could be reused many times over. Currently, sensor development requires that generally unique hardware and software be developed for their implementation. Sensors created through the ADAPT program would utilize more modular and reusable hardware and software; the driving factor for sensors, packaging, and components will be the needs of the mission that calls for their use. Through DARPA's research, researchers have developed a prototyped WSN capable of autonomous network formation, object tracking, and propagating track information throughout the network, while keeping the cost per node less than $1,500 [23].

Leading up to the ADAPT program, DARPA spent considerable time investing in WSN research, as evidenced by programs like SensIT, Smart Dust, NEST, and TinyOS [24]. Arguably, such research started with the distributed sensor network (DSN) program, which began in the 1980s [25].

3. Single Board Computers

As predicted by Moore's Law [26], the number of transistors able to be embedded onto a chip increased rapidly during the period beginning in the 1960s and continuing into the 21st century. In the early 2000s and into 2014, a number of individuals capitalized on the availability, affordability, and the shrinking size of processors to create tiny, single board computers (SBCs) and microcontrollers geared toward students and hobbyists. The first of such products, the Arduino, was released in 2005 by an Italian startup and meant for use by students learning electronics at the Interaction Design Institute Ivrea [27]. The Arduino was a 32-bit, 16-MHz microcontroller with 14 digital input/output pins [28] and could be purchased for approximately $30 U.S. [27]. All hardware and software were kept open-source [28]. What made the Arduino unique was not only its small size and low-cost, but also its ease of programming and ability to control a large variety of commercially-available sensors [29]. This made it a suitable candidate for potential inclusion into prototyped WSNs.

In 2012, the Raspberry Pi, an SBC with an Arm-Processor running Gnu/Linux also designed for students and hobbyists, entered the market selling for only $25 each [30]. The Raspberry Pi was the size of a credit card, had the ability to connect to a wide variety of peripherals to include Wi-Fi adapters and cameras, and had enough processing power to play high-definition video [31]. The Raspberry Pi proved to be extremely versatile and was utilized in a variety of applications to include webservers [32] and low-cost surveillance systems made of commercially-available components [33].

Other commercially manufactured SBCs capable of being networked and interfaced with sensors included the BeagleBone [34], the BananaPie [35], the Android MK802 [36], Via APC [37], and the Intel NUC [38]. Some companies combined microcontrollers, mini computers, and sensors into do-it-yourself kits for hobbyists—Ninja Blocks was one such company that combined the BeagleBone, Arduino, and a suite of various sensors into a kit for building custom, automated home security systems at a price of $199 [39]. The growing availability of SBCs and microcontrollers capable of being interfaced with cheap commercial sensors and various networking components offered exciting opportunities to rapidly build robust WSN prototypes, custom-tailored to satisfy fine-grained military use-cases, to include automating the LP/OP.

4. Node.js

Node.js was a software platform for creating networking applications. This framework was ideal for network programming, where multiple clients requested a service. Its strength was the speed in which requests were serviced. This speed was derived from its asynchronous nature, where requests were handled on a single-threaded event loop. Node.js was built on top of Google's V8 JavaScript engine, which compiled JavaScript instructions to machine code. A growing community of web developers and industry leaders migrated to this platform for the high performance Node.js offered [40]. Due to this increased following, the community developed a large code base of modules, which aided in the rapid development of networking applications. This technology is relevant to the problem of automating the LP/OP through the use of WSNs because of its ability to serve high performance real time applications that are suitable for use on mobile devices, thus enabling a robust user interface to the system.

5. Responsive Design

As the number of mobile, handheld, and tablet devices increase, developers/designers face an ongoing challenge to effectively deliver their web content to the user. In light of this shift toward using mobile devices to access web content, the initial instinct of developers was to design an independent "m.domain" site that would be served to all mobile devices [41]. Developers quickly realized the challenge in maintaining two separate web sites, as well as the significant challenge of getting the content to display correctly for any device, screen, operating system, and browser. Responsive web design (RWD) purports to address these problems and has garnered considerable attention from the web development community since 2010. This is an interesting concept for military applications, since all too often products are developed for the military utilizing proprietary technology/contracts that ultimately pigeonhole the military in a position that may not be the most beneficial solution for the long-term. This freedom of having a "one size fits all" mentality allows the application to be served to a wide variety of devices, giving commanders and troops the ability to decide what platform is best for the mission.

Ethan Marcotte first coined RWD on May 25, 2010, when he wrote a seminal article on the fundamentals of the concept [41]. The overarching idea can be best encapsulated by his statement, "This is our way forward. Rather than tailoring disconnected designs to each of an ever-increasing number of web devices, we can treat them as facets of the same experience. We can design for an optimal viewing experience, but embed standards-based technologies into our designs to make them not only more flexible, but more adaptive to the media that renders them. In short, we need to practice responsive web design" [41]. This article introduced the three building blocks of RWD as fluid grids, flexible images, and media queries. Through the use of these tools, in designing web applications, it has become possible to serve content to a wide variety of devices ranging from mobile to desktop, without losing the importance of the intent and content of that information—even when viewed on small screens.

6. Military Push for Mobile Device Applications

Though the United States military recognizes that there are still security vulnerabilities to overcome with wireless communication, it is accepted that the future of military computing will embrace the mobile domain [42]. "The Army Cellular Capability Development Strategy," released in 2011 by the U.S. Army Signal Center for Excellence, presents a vision where cellular technology is heavily utilized for battlefield communications, with smartphones being the workhorse for a multitude of combat tasks by different levels of leadership. Such a future strategy would "leverage commercial communications infrastructure for units both in garrison and while operationally deployed" [43]. The Army has even established a course for soldiers to learn how to write smartphone applications for military use [43].

While the Department of Defense has recognized the ubiquity of mobile devices and networking and has taken steps toward embracing the technology, it still faces significant challenges in implementation beyond security concerns. One of the biggest difficulties is the difference in devices that are used. There is not a single type of platform that the military has chosen to utilize at all times [43], and this multitude of devices complicates the development of new applications. Operators may utilize Android, iOS, or BlackBerry devices. Tablet and smartphone variations of these devices also exist, often with different screen sizes and device characteristics. These differences complicate the development of new applications, and it is a significant challenge to allow the widest degree of cross-compatibility among devices, especially since developing for these different platforms requires knowledge of different programming languages.

In order to reduce labor requirements for development and to allow the greatest cross-compatibility, code that can be written once and utilized across devices is highly desirable, and HTML5, an Internet markup specification, can possibly meet this need by providing a common development language [43].

D. SUMMARY

This chapter has shown how the application of wireless sensor technology to the tactical support of ground troops in combat has been limited. Existing sensor systems are too expensive for widespread deployment by infantry units, too complex to be utilized directly by the warfighter, and not capable of being monitored without specialized equipment that places a further weight burden on the already overloaded infantryman. This chapter has also highlighted key technology capable of providing a solution to this problem, showcasing the drive of the United States military toward the use of mobile devices for battlefield communications tasks. In Chapter III, the concept of a wireless sensor system for use by the infantry squad conducting an LP/OP is presented. This concept explores leveraging COTS components to build a cheap, effective, and highly usable system for increasing the situational awareness of infantrymen through computer automation.

III. MOBILE SITUATIONAL AWARENESS TOOL CONCEPT DESIGN

This chapter introduces the infantry task of conducting the LP/OP for unit security in combat. It discusses the challenges and limitations that units face when using this technique. Next, it closely examines Marine Corps requirements for conducting the LP/OP and then the historical lack of automation for LP/OPs, especially in light of advances in technology and the direction the armed forces are taking with respect to mobile computing. It then explores the requirements for automating the LP/OP, and presents a concept for an automated solution including the sensor nodes, the application server, and the mobile devices required for such a system. Following this, the chapter suggests a number of possible use-cases for such a system, with proposed tactics, techniques, and procedures (TTPs) for employment. Finally, it proposes how such a system could be incrementally tested as it is built.

A. PROBLEM DESCRIPTION

1. Utility of the LP/OP

The LP/OP has served the vital function of being the eyes and ears of combat leaders since the earliest days of human conflict, buying a force precious time to react to enemy maneuver by providing early warning of intrusion into a geographical area of interest [9]. The LP/OP offers a simple trading of time for space; by positioning troops forward of friendly lines with the task of watching for enemy movement, advanced knowledge of enemy activity is learned. This advanced knowledge then drives the combat leader’s own scheme of maneuver, allowing for the repositioning of personnel and weapon systems in order to more effectively engage the enemy. Additionally, because LP/OPs are positioned to best observe enemy troop movement, the LP/OP can potentially arrange indirect fires, such as artillery and mortar fire [44]. This requires that the LP/OP have personnel trained in the call for fire procedures (CFF) and have a real-time communications capability, either through radio or field phone, with the element providing fires [44]. Indirect fires have the added benefit of buying even more time for the combat force through disrupting and delaying a maneuvering enemy.

2. Limitations of the Traditional LP/OP

The utility of the LP/OP is apparent, but executing this tradeoff of time for space comes at a considerable cost to a combat force, and there are also several risks that the combat leader must mitigate. Operating LP/OPs requires considerable planning by involved leadership. Leadership must create a rotation schedule in order to keep LP/OPs continuously manned with fresh troops [7]. This rotation schedule is vital in order to ward off fatigue and boredom, which causes inattention by LP/OP operators and degrades the effectiveness of their position [45]. This inattention can lead to the operators themselves becoming vulnerable to sneak attacks by the enemy. The rotation schedule can be difficult to integrate into the overall security schedule of the main body of troops and is furthermore difficult to optimize for the greatest amount of rest. Adding more LP/OPs to the battle plan only compounds the complexity of the problem.

Regardless of the efficiency of the rotation schedule, LP/OPs demand a greater aggregate amount of work for the executing unit, which adds to combat stress, fatigue and risk. Since a combat unit must constantly maintain a ready posture, removing troops from the local security rotation to man the LP/OPs shortens rest cycles and more quickly fatigues the unit. In addition, moving back and forth between LP/OPs and main battle positions requires further exertion. This movement can also draw attention from the enemy and expose friendly forces to enemy fires. Further risk is caused by the threat of fratricide when forces are positioned forward of friendly battle positions, as is the case with the LP/OP. Mitigating potential fratricide requires additional planning—leaders must plan their fires with LP/OPs as a major consideration [7]. The presence of LP/OPs can potentially constrain the fires of friendly forces engaging the enemy if the LP/OP is positioned within the danger area of friendly fires, thereby limiting the combat power of the force.

In addition to the risk of fratricide, the LP/OP carries the additional risk of being cutoff from friendly forces. This risk is owed to the fact that the LP/OP is positioned forward of friendly units and is therefore more easily isolated. Being cut off from friendly troops and support could result in the destruction of the LP/OP or the capture of occupying personnel.

Finally, since the LP/OP manning is drawn from the main body of the friendly force, the combat power available to be massed on the enemy by that force is diminished.

These disadvantages exist for a common reason: human beings man the LP/OP. As human beings become fatigued they stop paying attention; they must therefore rest. Human beings are vulnerable to friendly and enemy fire, and can be captured by the enemy. Human beings must eat, drink, and eliminate waste, which can attract enemy attention. They also may become bored and demoralized when isolated for long periods of time, which degrades combat performance [46].

3. Formal Marine Corps LP/OP Requirements

In order to build a system that automated the functions of the LP/OP, it was necessary to first precisely define the functional requirements of the LP/OP. To do so, we examined relevant Marine Corps warfighting publications.

The Marine Corps Infantry T&R Manual precisely defines the requirement for the infantry fire team to conduct a LP/OP [7]. This task requires that the Marine infantry fire team "provide early warning while seeking to avoid direct enemy contact" as part of defensive operations, both day and night. This formal requirement states that this task must be accomplished while "supporting a defensive scheme of maneuver during daylight and limited visibility." Limited visibility, in terms of tactical situations, is meant to be any condition that would degrade the visual acuity of military personnel. Such conditions include darkness at night, adverse weather conditions (e.g., rain, snow, sandstorms), or observation-limiting terrain such as heavy vegetation, dense urban structures, or micro terrain (e.g., hills, draws, valleys, or depressions) that interfere with line of sight [7].

According to this standard, executing the LP/OP consists of many critical components. The LP/OP must be planned; it must be resupplied; and it must maintain communication with higher headquarters. It should also report critical information when observed. It must maintain its own security and there should be a plan to break contact with the enemy and move back to friendly defensive lines, which requires coordination for the passage of friendly lines [7].

MCWP 3-11.3, *Scouting and Patrolling* [47], an authoritative Marine Corps publication on small-unit infantry operations, provides further guidance on the proper execution of the LP/OP. In choosing the location of the LP/OP, the site should not be prominent and therefore obvious to enemy forces [47]. OPs should also not be manned for more than 24 hours; however, in practice they are frequently manned for longer periods of time due to the work required in constructing a new position. LP/OPs should be occupied and exited using diverse and concealed routes. Observers in the LP/OP should utilize all pertinent senses, to include sight, smell, and hearing, to discover and report enemy activity.

The information to be reported regarding the enemy by the observers is a critical component of every LP/OPs mission. Every situation will dictate independently what unit leaders want observed and reported. The NATO spot report (SPOTREP) provides a standard used by Marines to report enemy activity [47]. The SPOTREP describes the size and strength of the enemy unit being observed, the actions of this enemy, their location, the enemy unit (identification), the time of the enemy observation, and the equipment and weapons being utilized by the enemy collectively referred to as Size/Activity/Location/Unit/Time/Equipment or SALUTE. This acronym, SALUTE, is widely used at the small-unit level whenever there is a need to thoroughly describe the nature of the current enemy [47], such as when a unit leader briefs his Marines in an operation order on the nature of the enemy they will face.

MCWP 3-11.2, *Marine Rifle Squad*, also provides information on the LP/OP [48]. This source gives a metric on the distance from friendly lines that an LP/OP should be employed while the unit is in a security posture. The prescribed distance is 460 meters, which is approximately the maximum effective range of the M16/M4 battle rifle employed by the majority of infantry Marines.

4. Historical Lack of Automation

The LP/OP has undergone minimal automation despite often being supplemented by tools that allow enhanced perceptual abilities [47], such as telescopes and infrared optics. These supplements have not significantly reduced the human involvement in the conduct of the LP/OP, as most continue to require perpetual human monitoring.

As discussed in Chapter II, the United States military is embracing the ubiquity of mobile computing in the civilian sector and exploring methods of adopting this technology to automate various battlefield tasks. Local surveillance and security, the kind provided by troops conducting LP/OPs, should be one of these tasks aided by the growth of mobile device use within the military.

5. Requirements for an Automated LP/OP

We utilized authoritative Marine Corps publications for identifying the primary requirements of a system that would automate the LP/OP. The system should be capable of detecting an enemy in sufficient detail so as to satisfy the requirements of the SPOTREP and SALUTE; it must be able to do so both during daylight and in low-light conditions. To accomplish this task, the system should be capable of utilizing multiple sensing modalities similar to the way Marines use multiple senses to detect hostile activity. Additionally, the system should be able to report the SPOTREP in near real-time to the employing unit leader so as to provide actionable information. Furthermore, the system needs to be inconspicuous so as to avoid enemy detection.

We identified further critical requirements for designing an automated LP/OP system, not found in the Marine Corps formal publications. First, the system would have to be cost effective to be fielded in quantities large enough for widespread use throughout the Marine Corps' infantry units. The strategy for achieving this low per unit cost would be to utilize COTS rather than building expensive, specially designed hardware and software.

Second, the system would need to be easy to learn and utilize, requiring little time for the average Marine rifleman to become proficient with its employment. Infantrymen must learn and maintain proficiency in a wide variety of skills, and therefore extra time for training is scarce. Most infantrymen have little training in sensor system employment and operation; this system should not place an extra training burden on the infantry unit. Ease of use would also ensure the ability for the widest possible group of infantrymen to use the system, thus enabling great flexibility in its employment. In order to accomplish the highest level of usability, the user interface (UI) of the system would need to leverage the widespread familiarity with commercial mobile devices that has accompanied the boom in smart devices (i.e., tablets and smartphones) and the accompanying popularity of such devices amongst young service members [43]. Therefore, the design should mirror the "look and feel" of applications commonly found in the civilian market and utilized by Marines in their personal lives. Also, with the predicted proliferation of mobile devices on the battlefield [43], the system should be capable of interfacing with the widest variety of mobile devices.

Third, in order to network the COTS components of the system together effectively, we determined that system components should utilize the ubiquitous Transmission Control Protocol/Internet Protocol (TCP/IP). This would allow connecting the widest possible variety of parts and the possibility of later augmenting or substituting pieces of the system with other IP-compatible sensors or components, as well as incorporating new types of handheld monitoring devices (i.e., new models of smartphones and tablets). Also, utilizing TCP/IP and COTS parts would allow for the fastest development of a prototyped system, the ability to make timely improvements, and for the rapid adaption of the system to satisfy changing mission requirements.

Fourth, the system as a whole should be practical for dismounted infantrymen to carry and employ. With the average infantrymen carrying more than 60 pounds of gear in his combat kit [49], the system would have to be as light and portable as possible. Due to constraints on power, the system should also have its own power source; weight limitations would limit the amount of power available to the system. The system should have enough power to have a practical life span in the field, meaning it should be able to be continuously operated for a significant number of days with only its onboard power source.

Finally, in order to be practical for use by the warfighter in stressful combat situations, the system should be as autonomous in operation as possible, meaning very minimal effort would need to be required on the part of the human operator to initialize the sensor network and maintain its useful operation. A self-configuring, ad-hoc network of sensor nodes was therefore preferred. In short, the system should be easily networked, cost-effective, limited learning curve, easy to use, extremely portable, long endurance, highly autonomous, and have an extensive effective field of view.

B. CONCEPT FOR AN AUTOMATED SOLUTION TO THE LP/OP

1. Problem Solution: Proposed Architecture

Here, we propose the MSAT, a system for automating the LP/OP thereby reducing the manpower burden for an employing unit. This system would leverage powerful modern computing technology, and should be able to be implemented at the lowest levels of combat units within the military due to its low cost and ease of use. Such a system consists of sensor nodes, an application server, and mobile devices for interacting with the system.

a. Sensor Nodes

To build MSAT, a sensor node is needed that can perform as the backbone of the system, acting as the primary means of alerting the operator to enemy activity. The sensor node itself must satisfy several specific criteria in order to be a viable option for incorporation into the system. First, the sensor node must contain sensor modalities capable of intrusion detection into a tactical area of operations (TAOR) and target classification upon detection specific enough to satisfy the requirements of the SPOTREP. We determined, through research of modern sensor system implementations, early in the planning of the system’s design that the most useful sensors would likely be passive infrared (PIR) sensors, cameras, seismic sensors, and acoustic sensors, as well as a global positioning system (GPS) receiver for determining sensor location.

Second, a system that already had a threat classification algorithm in place (i.e., the ability to track targets through the sensor field) would be preferred. Furthermore, processing should be done locally (i.e., onboard the sensor nodes) rather than centrally, in order to support expanding the number of nodes without putting strain on the entire network due to increased packet transfer and latency due to insufficient bandwidth across the network. A scalable network that can survive the destruction and addition of nodes, without disruption and failure, is important when considering combat applications.

Third, the nodes would need to be capable of reliably communicating tracking information to the system operator utilizing an intuitive UI. To do so, the sensor node would need to utilize the TCP/IP protocol to transfer data packets, thus enabling threat and tracking information to be transferred to a monitoring device, in the hands of an infantryman, that displays the sensor field updates on a suitably designed UI.

Fourth, the sensor nodes would need to be capable of autonomous network formation for ease of use in combat. Network formation and operation should be decentralized in order to increase the survivability of the system in the event of destruction by hostile forces. Further, the network of sensors should be as impervious of exploitation by the enemy as possible should nodes be captured or the network interconnecting the nodes by detected and monitored by the enemy.

Fifth, the overall system would need to be inexpensive enough to be fielded in large numbers by infantry troops and in quantities great enough to enable useful battlefield automation down to the tactical level. The sensor node itself would also need to be inexpensive, preferably significantly cheaper than the purpose-built sensors currently used by military.

Finally, the nodes would need to be light enough to be carried over long distances by dismounted infantry, who themselves would likely be carrying heavy personal combat loads. They would also have to utilize wireless communications so as not to force the warfighter to manage unwieldy cables between nodes. The nodes would also need to utilize an internal power source and have a life-cycle of at least fourteen days in order to be utilized tactically in combat.

In light of these criteria, a variety of options were examined for possible use. The first was creating custom sensor nodes that would satisfy the above requirements utilizing commercial SBCs (e.g., RaspberryPi, BeagleBone), microcontrollers, and different types of sensors. This method would require intensive hardware design and engineering and the creation of complex communication protocols that would enable the nodes to establish an ad-hoc wireless network and autonomously share information between nodes. Though it was believed possible, the level of complexity required for these tasks was determined to be outside the scope of this thesis. Thus, existing hardware requiring little modification would be required for use in the MSAT system.

In summary, we sought a sensor node system that incorporated multiple onboard sensor modalities to perform a locally executed threat classification algorithm, capable of autonomous network formation and decentralized operation, with the ability to interface with a variety of handheld devices through the ubiquitous TCP/IP protocol. Additionally, the sensors needed to be low-cost for widespread use throughout the military, lightweight to facilitate portability, and wireless in order to be practical. Finally, the sensors should have an onboard power source making them capable of operating in the field for 14 days.

b. Mobile Devices

The United States military has increasingly used mobile devices in training and operations, and the Army plans for these multi-purpose personal handhelds (e.g., smartphones and tablets) to take a central role in battlefield communications in the future. There has been a movement to develop many different applications for use on these devices to satisfy varying mission requirements [43]. Though the military recognized the utility of mobile devices and the importance of developing applications for these devices, they did not decide upon a single model, make, or operating system to employ in all cases. Therefore, running the same software application across a diversity of devices presents itself as a major challenge. This challenge is due to the fact that different types of devices are programmed in different programming languages—for example, most native Android applications are coded in Java, while most iOS applications are coded in C# [43]. Also complicating the task, devices may have a different operating system, or operating system version, and hardware characteristics that make running a universal application complex.

Recognizing the multitude of different devices on the market today and in use across the Department of Defense, we decided that we wanted the MSAT system to be compatible with the widest range of devices. Due to hardware and operating system differences already identified, the task of creating separate native applications for the myriad of devices was determined to be an impractical approach for our research. Instead, we wanted to be able to write our program once and have it run on the majority of mobile devices. “Write once, run anywhere” is a commonly referred to goal in software engineering and the introduction of HyperText Markup Language 5 (HTML5) is purported to assist web developers in achieving that goal [50]. To promote interoperability between the widest variety of mobile devices and the sensor network a web-based user interface was proposed. A web-based front-end would enable the greatest possible code reuse and achieve compatibility with both Apple and Android devices while limiting the impact on development manpower resources. This cross-platform capability comes from standards implemented across Internet browsers that exist on mobile computers [43]. Web applications written in HTML5 have the capability for both online and offline application operations [43]. The overall layout proposed is a web-based interface written with HTML5, utilizing both JavaScript for functionality and cascading style sheets (CSS3) for styling. Based on our research of web development and the ease of accessibility to online communities and resources, we decided that a web application would be the most appropriate design for MSAT.

The user interface to the system would need to enable the greatest possible usability for operators in stressful combat situations, and provide the greatest degree of situational awareness. A usable system for our military use case would be appropriately defined as, “the effectiveness, efficiency, and satisfaction with which specified users can achieve specified goals in a particular environment” [51]. This would need to be thoughtfully considered throughout the developmental process, so that the focus would be maintained on the end user, the infantrymen, who would utilize the tool.

The user interface to MSAT should facilitate the greatest degree of situational awareness possible on the battlefield. To enable this, the main component of the interface would need to be a map. The map should be capable of being scrolled, zoomed, and rotated, similar to the capabilities of a variety of mapping applications available on the Internet today. Next, the interface would need to overlay sensor nodes on the map. Finally, the interface should give alerts to the operator of threats in the sensor field. To communicate intrusions into the sensor field, the interface would need to display visual indicators of intrusions, such as flashing graphics where intrusions occur, or lines displaying the tracks of targets moving through the area. Redundant alerts would enhance monitoring capabilities; thus, audible alarms and vibration would be incorporated whenever possible, with the option of turning these features off if the tactical situation called for their silence. To allow threat classification through use of the system, a pop-up window would provide video or still images of intruders, in addition to other information regarding the target provided textually that the system can possibly provide using its multiple sensors and threat classification algorithms.

Additional features of the user interface would contribute to enhancing situational awareness. The system would track and plot friendly positions on the map graphically. The system would also provide chat functionality for communicating with adjacent team members and units. This would be useful for communicating enemy information gleaned from the system to friendly forces.

Figure 2 depicts a proposed wire/straw-man frame of what a usable interface would look like for the MSAT system, utilizing all the concepts that were previously discussed as requirements for the system.

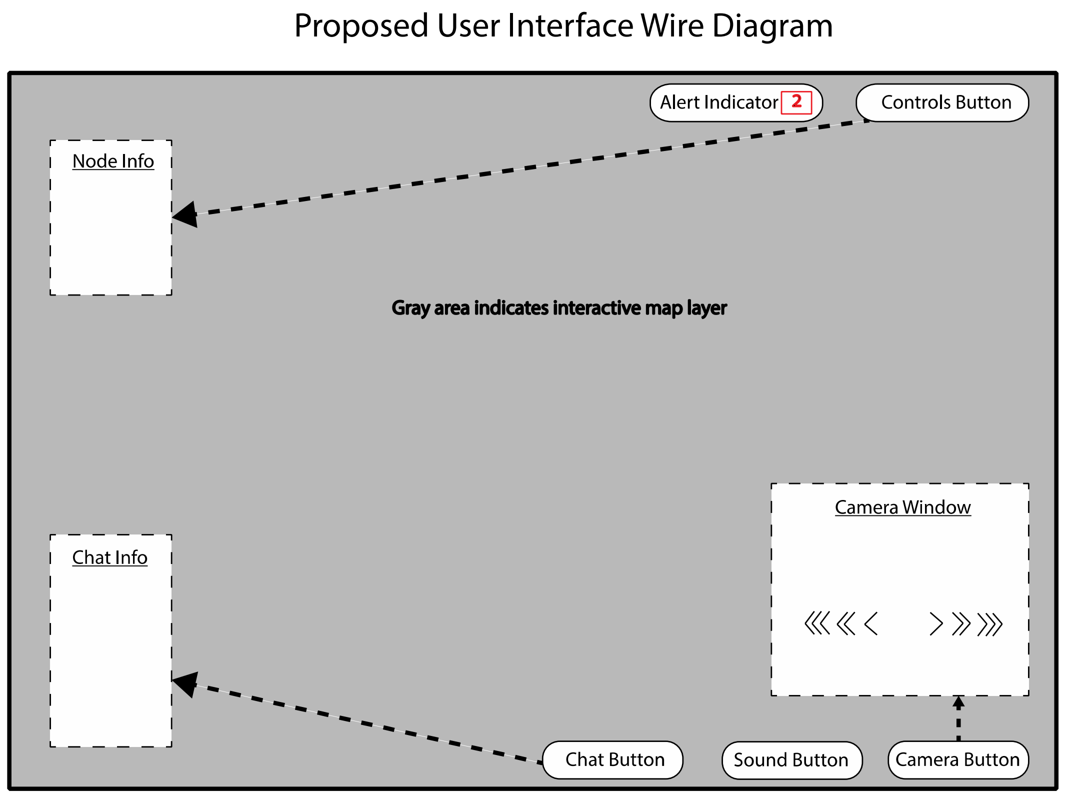


Figure 2. UI Wire Frame

The wire frame aids a web designer in articulating a vision of what the product may look like. Our focus is on maximizing the real estate of the map layer (i.e., gray background) and to keep reoccurring tasks, integrated as buttons, in the corners of the web application so that they are easily accessible by the user and not a hindrance to the users' observation of the sensor field. Once a button is triggered, the resulting action is imitated by a dashed line with an arrow, which shows where the pop-up container will populate on the screen and the title of its content.

c. Application Server

In order to facilitate the write once, run anywhere philosophy of the web application written in HTML5, JavaScript, and CSS3, an application server would have to be implemented through the use of an already existing web server application code base or the creation of a custom one. Another anticipated requirement is that in order to serve the application's dynamic content to user devices, an application server must be capable of receiving and responding to Asynchronous JavaScript and XML (AJAX) requests. AJAX can be best viewed as a group of tools that allows asynchronously updating user web pages. These tools include: standards based presentation through CSS3, dynamic display and interaction through the Document Object Model (DOM), data interchange and manipulation using XML (Extensible Markup Language), asynchronous data requests through the XMLHttpRequest object, and JavaScript which brings all these together through its multi-paradigm programming language [52].

d. *Overall System Design*

Figures 3 and 4 illustrate the proposed logical and physical network topology.

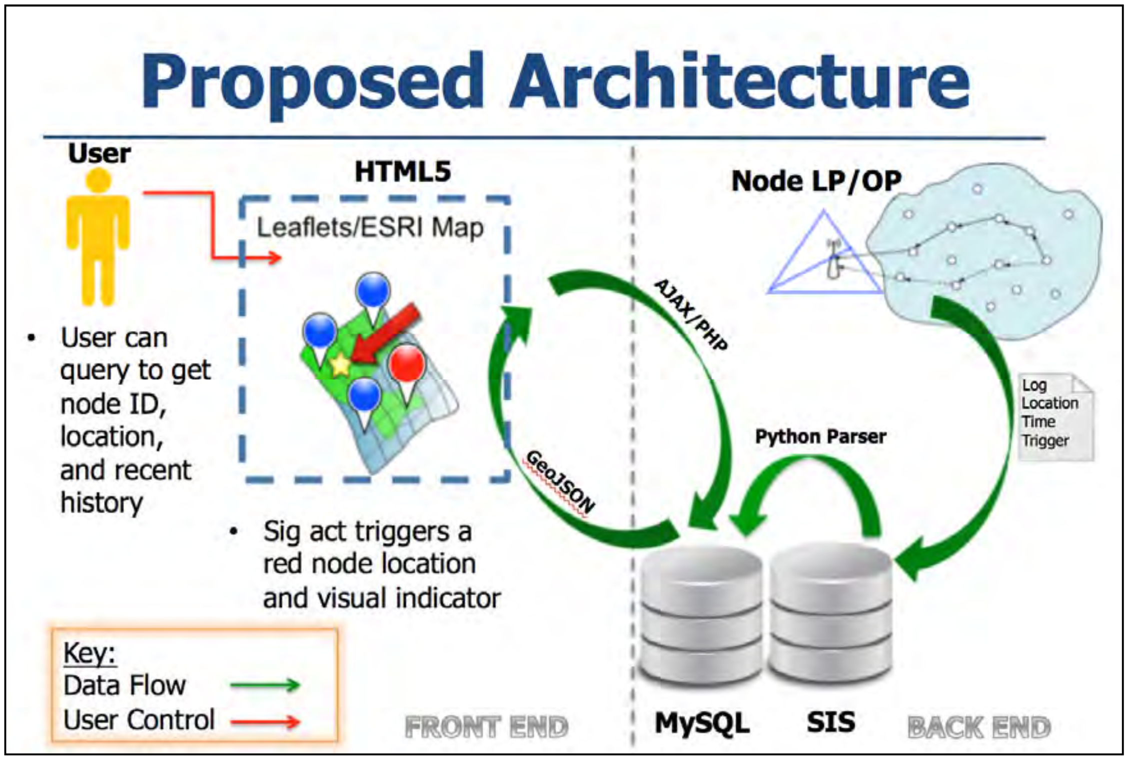


Figure 3. Logical Network Topology

The logical topology depicted in Figure 3 illustrates the data flow that occurs from the sensor field, through the application server, and ultimately to the client device.

This is a high level view that suggests what possible technologies that may be applied to the MSAT system.

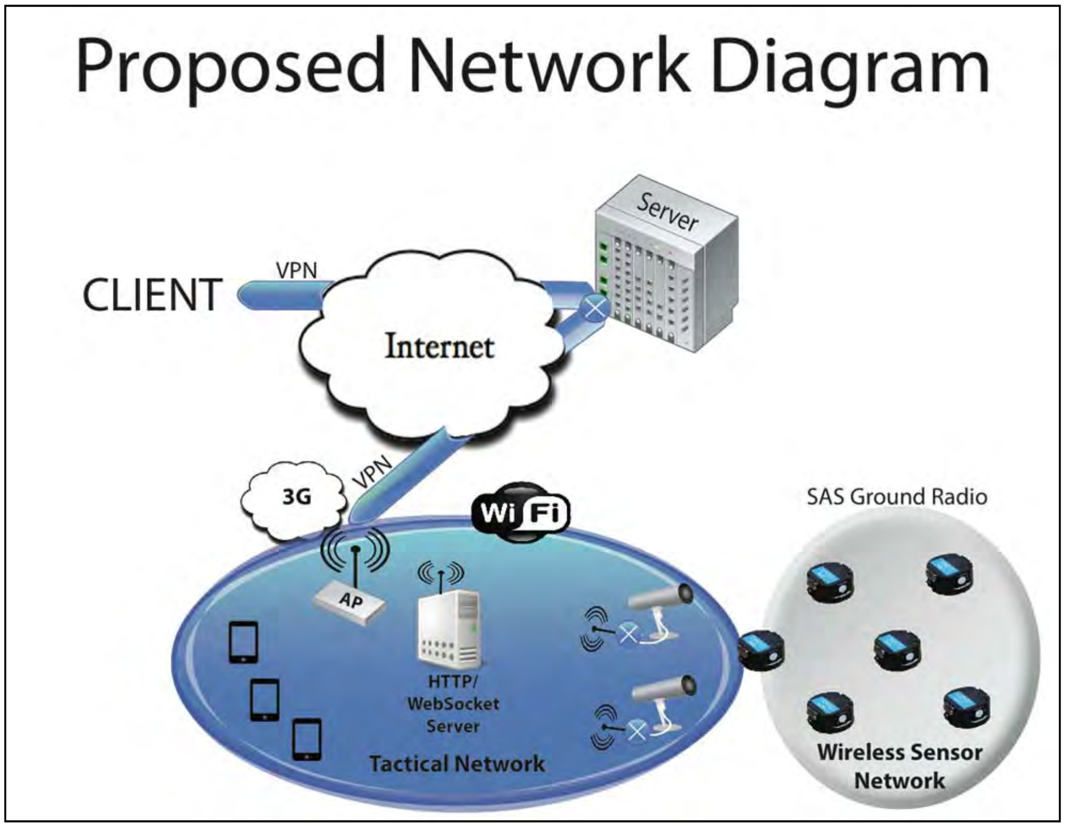


Figure 4. Physical Network Topology

The physical topology depicted in Figure 4 highlights the interconnections that may occur from the sensor field, through the application server, and on to the tactical client's device or to the remote client's device. This, like the logical topology diagram, is a high-level view depiction of the technologies proposed for the architecture of the MSAT system.

2. Possible Use Case and Employment

Here, we present several potential use-cases for the proposed MSAT system, covering a diversity of common combat scenarios faced by infantry units.

a. Ambush

Utilized by an infantry unit conducting an ambush, MSAT would provide early warning of an approach, into the sensor field. It would also provide the direction of approach and possible disposition, depending on the number of discernable tracks created. This would aid in the prevention of the erroneous triggering of ambushes, by neutral or friendly forces, through early notification of an intrusion in the sensor field and allowing the human to make a classification, via camera, earlier.

Utilized in an ambush role, as shown in Figure 5, sensor nodes would be placed as far out as possible on the flanks of the ambush position to facilitate the earliest warning of enemy presence. Additionally, nodes would be placed in the staging area, doctrinally known as the operational rally point (ORP), where extra equipment is often dropped by the unit and is the site for linkup after actions on the objective. The ORP may be left completely unmanned, and sensor nodes would give warning of enemy compromise of the position.

Ambushes can require units to occupy ambush positions for long periods of time while waiting for an enemy whose time of arrival is uncertain. This can result in boredom and complacency that leaves a unit unprepared when the enemy does arrive. Sensor nodes that alert personnel to enemy presence early through a user interface on a handheld device can refocus the attention of the entire unit, and use of the chat can allow for last minute coordination within the unit without breaking the silence an ambush necessitates.

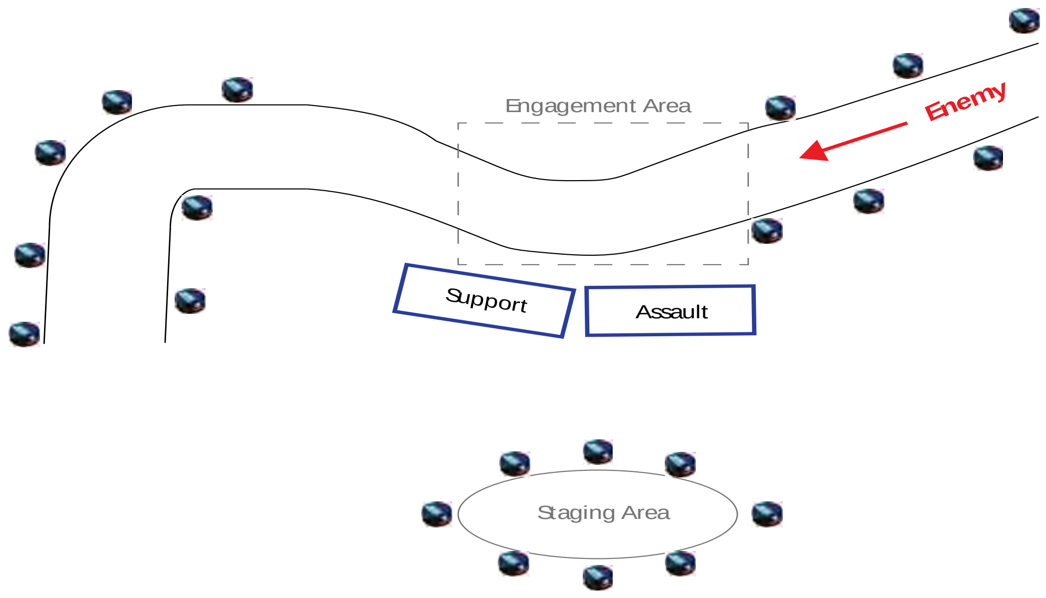


Figure 5. Ambush Scenario

b. Reverse Slope Defense

A reverse slope defense utilizes the concept of mass surprise fires to destroy the enemy by drawing him into an engagement area that silhouettes him against the slope of a hill after crossing an inter-visual (IV) line or ridgeline. This IV line (shown in the gray lined area) prevents the unit conducting the defense from visually observing the enemy before he enters the engagement area. To provide early warning of an advancing enemy, sensor nodes can be placed on the other side of the IV line. This avoids the necessity of placing personnel forward of the IV line and beyond supporting arms range of friendly forces, and possibly also masking friendly fires upon enemy contact. Three squad battle positions are depicted in Figure 6. They are focused on engagement areas, which are then mutually supported by sensors.

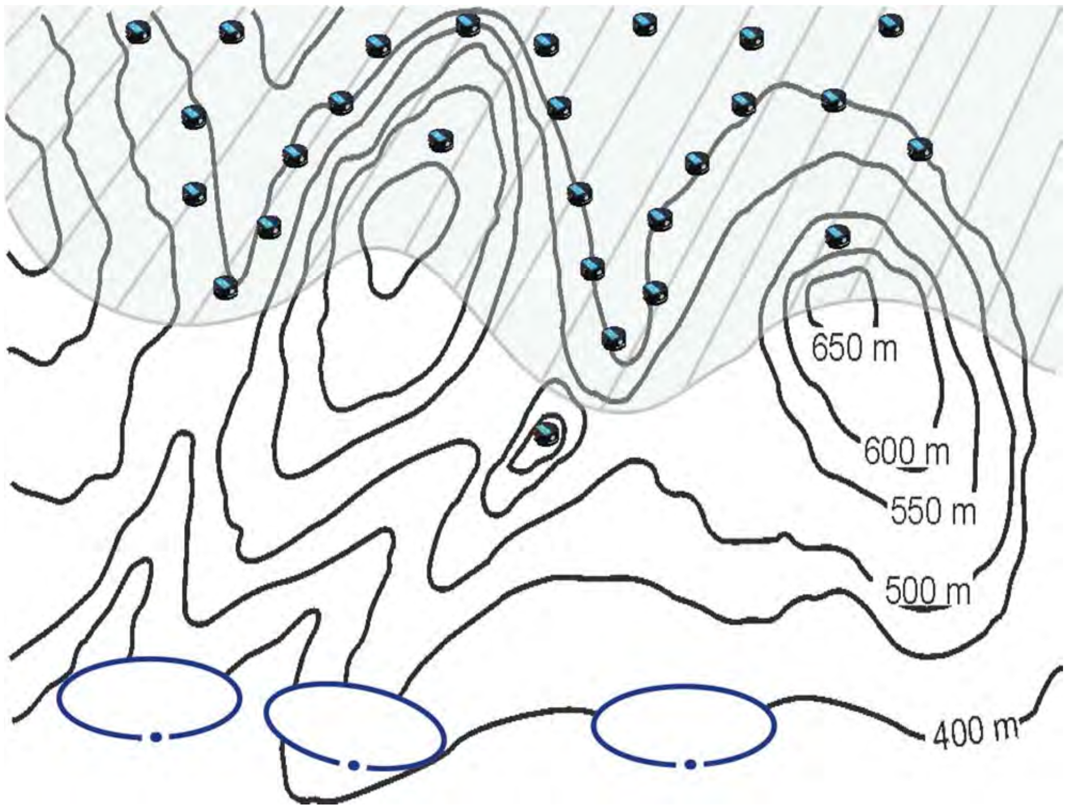


Figure 6. Reverse Slope Defense Scenario

c. Urban Defense

Small units can establish defensive positions in urban environments. Such environments are characterized by tightly-packed, multi-storied structures that create covered and concealed avenues of approach. Alleyways, sewers, trenches, rubble from damaged buildings, and rooftops can all be utilized by a clever enemy to safely close with friendly forces, where fragmentation grenades can then be thrown into friendly positions without ever having to expose themselves directly to small-arms fire. Sensor nodes could be littered, possibly covertly with special operations forces or overtly using indirect fire techniques leveraging the robustness of the UGS device design, through these areas to give forewarning of such enemy infiltration before coming within hand-grenade range. Shown in Figure 7 is a platoon battle position, in an urban environment, that is supporting its position by employing the sensors in the surrounding dead space (i.e., unobservable from the friendly position).

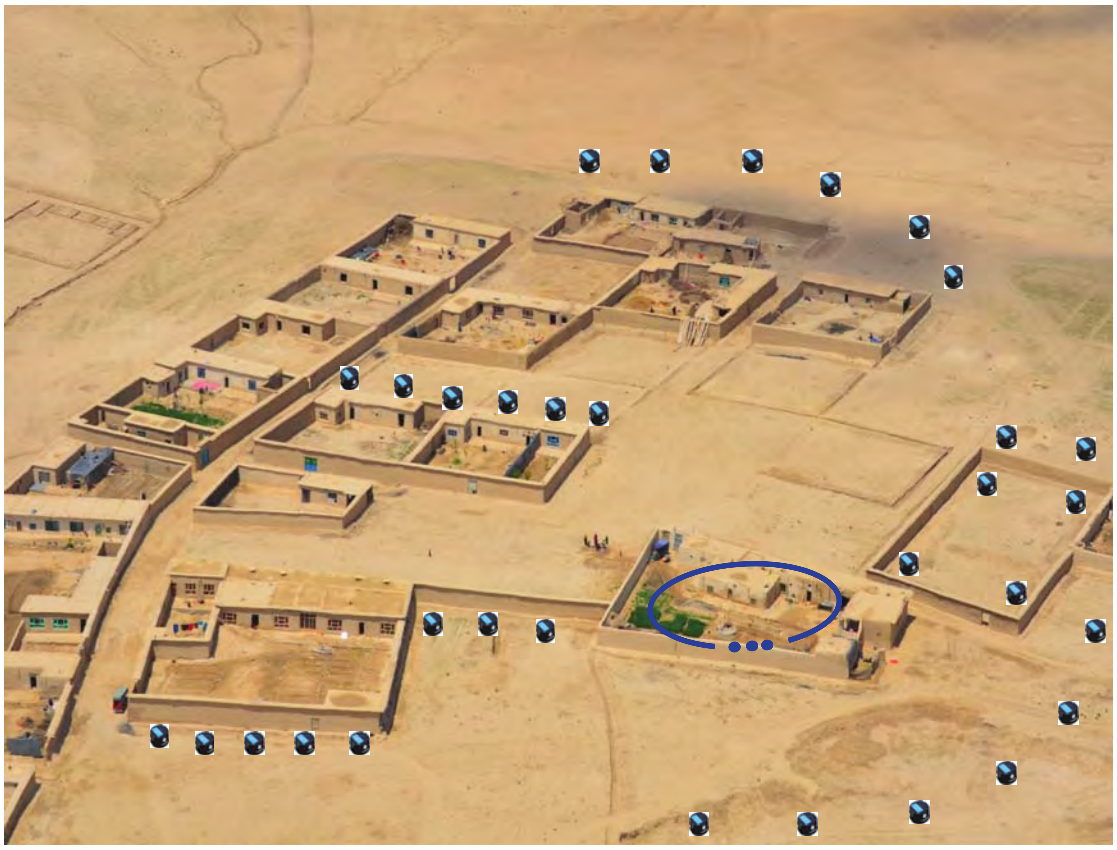


Figure 7. Urban Defense Scenario

d. *Tactics, Techniques, and Procedures for Employment*

In order to effectively employ the sensor field, unit leaders will conduct a reconnaissance of the area, prioritizing node placement based on dead space locations (i.e., unobservable terrain), likely avenues of approach, and natural lines of drift (i.e., terrain's tendency to influence movement). Patrols would then be dispatched to emplace the sensor nodes in their desired locations. Patrols should be organized into two elements—the node emplacing team and the security element. The node emplacing team will consist of at least a buddy pair—one individual carries the sensor nodes in a load-bearing pack, while the other retrieves the nodes from the pack and actually emplaces them. The security element provides protection and over-watch for the emplacing team, preventing enemy interference with the node emplacement.

3. Testing Plan

We use an incremental approach to testing the MSAT system, where the individual components of the system are tested, as they are developed. First we focus on the sensor nodes' ability to reliably detect and classify threats and then transfer that data from the sensor field. Tests are conducted to determine the reliable range in which the nodes can communicate to each other, to develop repeatable deployments of the system. The sensor field is tested with various wireless access points, which will provide a means to back haul the information from the sensor field to the application server. Testing is conducted with sensor field deployment in different environments (e.g., overhead cover, thick vegetation, light vegetation, pavement, low grass, and tall grass), to determine best practices for employing the devices and identify impacts on the system.

Second, we test the application server's ability to process the node information, parse the information into a transferable data packet, and handle multiple connections to clients. The application server is tested, in a lab setting, to determine capabilities of throughput, as well as the amount of concurrent client connections that can be handled reliably. The application server is tested as part of the MSAT system, in a field environment.

Third, we test the user interface and responsive design on multiple devices. The design is tested with various browsers, devices, and operating systems. This testing is not meant to be exhaustive of all devices, due to time and monetary limitations, but is intended to show the capabilities of a RWD that is developed with mobile platforms in mind.

Finally, we conduct a live field experiment, utilizing the system as a whole. This will provide an opportunity for operators to employ the system in a military scenario, with the objective being to determine when a person has entered the sensor field and to provide a SALUTE report based off the observations.

C. CONCLUSION

In this chapter, we discussed the LP/OP—its uses, its limitations, and its requirements as set forth by the Marine Corps. We showed how the LP/OP is a battlefield task well-suited for automation, and how the United States military's growing interest in mobile devices on the battlefield can be leveraged to create a flexible tool for infantry units to conduct surveillance and intrusion detection on the battlefield. The concept for such a tool, which we coined MSAT, was presented, including the system components and architecture. Finally, several possible use-cases and methods of employment were discussed, and a testing plan was laid out. In the following chapter, we present the actual development of the reference design for MSAT, the detailed test plan based on the reference design and test results of its individual components, and the performance of the system in a field experiment.

IV. MSAT IMPLEMENTATION AND TESTING

To show the possibility of using computer systems to automate the LP/OP for small infantry units, we built a functioning prototype, MSAT, consisting of COTS components, existing sensor nodes, and custom-built application software. We first selected an appropriate sensor node to act as the backbone of the system, and then we tested the capabilities of these sensors before incorporating them into a system that satisfied our use-case. Next, we built a software application that allowed for warfighters to interface with the sensors, enabling them to utilize the sensors for surveillance, intrusion detection, and target classification. We then tested the capabilities of this software isolated from the rest of the system. Finally, we built a mobile tactical network that integrated all of the components into a complete system for field use, and we tested the capabilities of the system in a mock combat scenario.

A. SENSOR NODES

We needed to incorporate sensors into our prototype in order to detect intrusions in the tactical area of operations (TAOR) and classify these intrusions, alert human operators using the system and enable them to use the system to gain battlefield situational awareness. In order to most effectively execute this task, multiple modalities of sensors would be required, and the sensors would need to be able to communicate environmental events wirelessly and in near real-time, with minimum effort or training required on the part of the operator. WSN nodes could accomplish all of this with multiple onboard sensors capable of autonomous network formation and a protocol for sharing gathered environmental information across the network, such that the information could be processed and presented in a human-readable format through a user-interface. Such sensors would serve as the backbone of MSAT.

Building such a sensor node from scratch would be too complex a problem for the scope of this thesis. Therefore, we sought an existing solution. The ADAPTable sensor system (ADAPT) Smart Munition prototype, built by DARPA, provided an acceptable solution and was therefore utilized as MSAT's sensor node.

1. ADAPT Smart Munitions

The ADAPT Smart Munition was born out of ADAPT, a DARPA program that began in 2012 with the purpose of building UGS systems of COTS technology that could serve as an intelligent replacement to conventional munitions, such as cluster bombs and minefields [53]. Such nodes would be capable of autonomous network formation and communication with command and control (C2) assets, after being hand placed, airdropped *en masse,* or delivered via artillery fire. We decided to utilize this prototype sensor in a manner other than originally intended by DARPA, by integrating them into our system and using it as the key technology of MSAT to directly support the infantry warfighter at the tactical level for surveillance purposes.

2. ADAPT Sensor Node Prototype Specifications

a. Core Hardware

The ADAPT Sensor node was built around the ADAPT core, a breadboard containing a Qualcomm MSM8960 Snapdragon System-on-a-Chip Dual Core processor, graphics processing unit, GPS Processor, and 3G/4G modem [53]. Cellular, GPS, Wi-Fi, and Bluetooth antennas were also integrated into the ADAPT core. The processor is capable of running at 1.5 GHz per core [54]. The ADAPT board utilizes a removable MicroSD card for persistent storage [53]. Various views of the sensor node are provided in Figure 8.

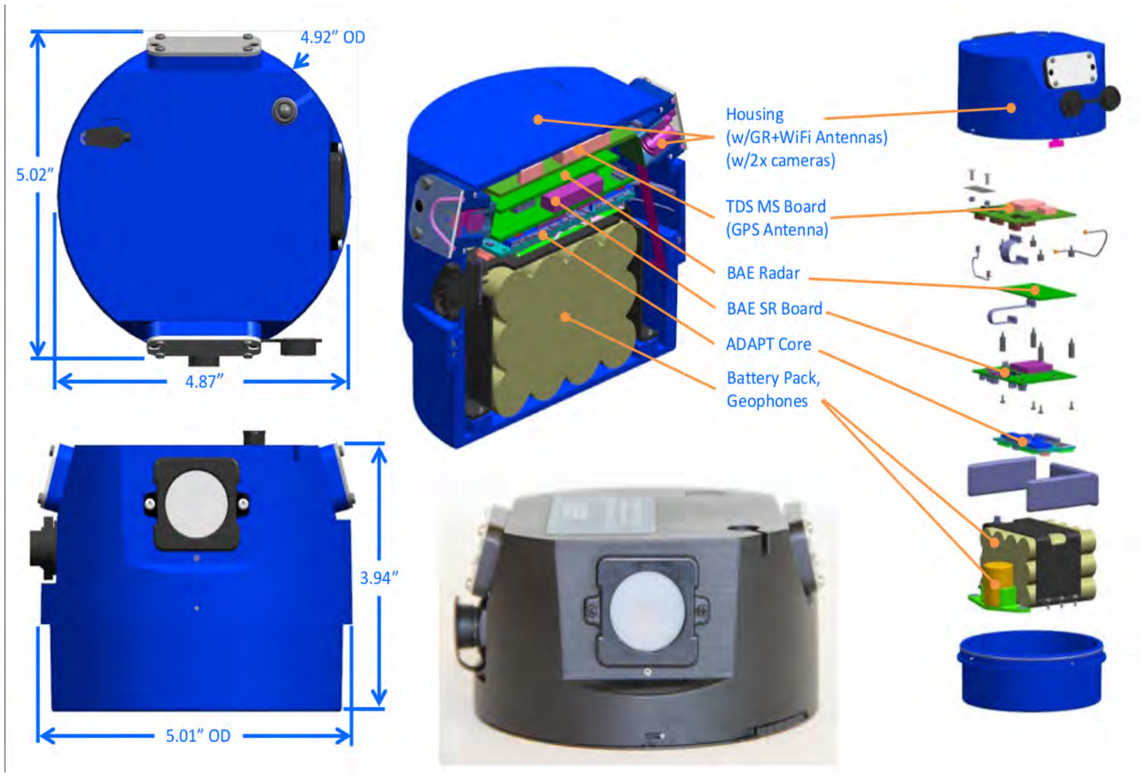


Figure 8. ADAPT Sensor Top, Side, and Cutaway Views, from [53]

b. *Sensors*

The ADAPT node contained a collection of different sensors. Each node utilized a PIR positioned at the front of the housing, capable of detecting intruders from a range of a few meters out to approximately 50 meters for larger targets such as a vehicles, and out to 20 meters for smaller targets such as human traffic.

Additionally the ADAPT sensor node contained a physical tripwire. The tripwire was held in place magnetically to the side of the node housing, and the wire, which was made out of string, could then be unraveled to its desired length. The tripwire system was designed so that when the tripwire was disturbed, the magnet connected to the end of the wire would pull apart from the magnet on the node, and the actuator would trip, causing a detection event on the node.

The ADAPT sensor node also contained two seismic sensors, an acoustic sensor, and two cameras mounted on opposite sides of the sensor node. The cameras were intended to be utilized for the capturing of still images and video.

c. Housing

The ADAPT sensor node consisted of a cylindrical, hard plastic housing with a radius of approximately 2.5 inches and a height of approximately 4 inches. Rubber seals were used at all of the connections, where the plastic components met, in order to make the nodes water-resistant.

d. Operating System

Each ADAPT node ran a version of the Android operating system custom built for the ADAPT program, optimized for low-power consumption in order to increase the operating lifecycle of the nodes. This modified, headless Android operating system was built on top of the Linux kernel, and utilized components built with the C programming language [53].

e. Node Software

Software running on the operating system of the ADAPT sensor nodes enabled the key functionality. The Shared Information Space (SIS) process enabled data sharing between nodes, through the running of a small-footprint database. The Scheduler and synchronous/asynchronous (SAS) medium access protocol processes were responsible for ground radio communication between nodes (i.e., the medium access control (MAC) policy). The RTC Real Time and System Clock Synchronization was a process responsible for clock management. Other applications were responsible for other key tasks: *loc* computed node locations, *pir* was responsible for running the PIR sensor while *libpir.so* handled detection data from PIR-triggered events, *seismic* processed events from the seismic sensors, and *libtrack.so* handled tracking data.

Additionally, the operating system contained drivers for the field programmable gate array (FPGA), the 900 MHz ground radio, the geophone, and the PIR sensors. An important note is that a functioning driver for the video cameras on the nodes did not exist [53].

f. Power Consumption and Duty Cycle

The ADAPT nodes were powered by rechargeable lithium-ion battery packs housed at the bottom of every node. The Android operating system allowed for power-aware operation, and the management of different power states in an effort to conserve energy. The ADAPT node operated with six different states of power usage.

The nodes were intended to spend most of their time in the vigilant power state, defined as having the PIR sensor enabled, the processor sleeping, and the ground radio listening on only one receive slot. In this state, the nodes simply wait for an intruder to enter the field and listen for events from their neighbors. While doing so, the nodes consumed 80 mW and were capable of operating for an estimated 62.5 days.

In the characterization state, the node had its PIR sensor on and also turned on its seismic software for classifying threats. This more-aware state was entered when neighbor nodes transmitted knowledge of local intrusions so that the sensor node could be ready to more quickly detect and characterize threats. In this state, the sensor node was also listening on one receive slot. The sensor nodes consumed 412 mW and could operate an estimated 12.1 days continuously in this state.

The nodes entered the tracking state after the PIR sensor was tripped by an intrusion. In this state, all of the sensors were turned on, Wi-Fi was enabled in order to relay information directly back to the monitoring station, and the ground radio broadcast slots were utilized in order to transmit detection and tracking information directly to neighbors via the SAS protocol, discussed in the section below. In tracking mode, the ADAPT nodes utilized 568 mW of power and could operate for an estimated 8.8 days continuously.

On the node’s initial startup, it would enter the GPS-on state. This state would stay active for one hour, to allow for the network to stabilize and acquire locations for all the neighbor nodes. Periodically, the node would activate its GPS-on state for ten minutes, every four hours. In this state, nodes used 804 mW of power. If the nodes were in a GPS denied environment, they would derive a center of mass calculation from neighbor nodes.

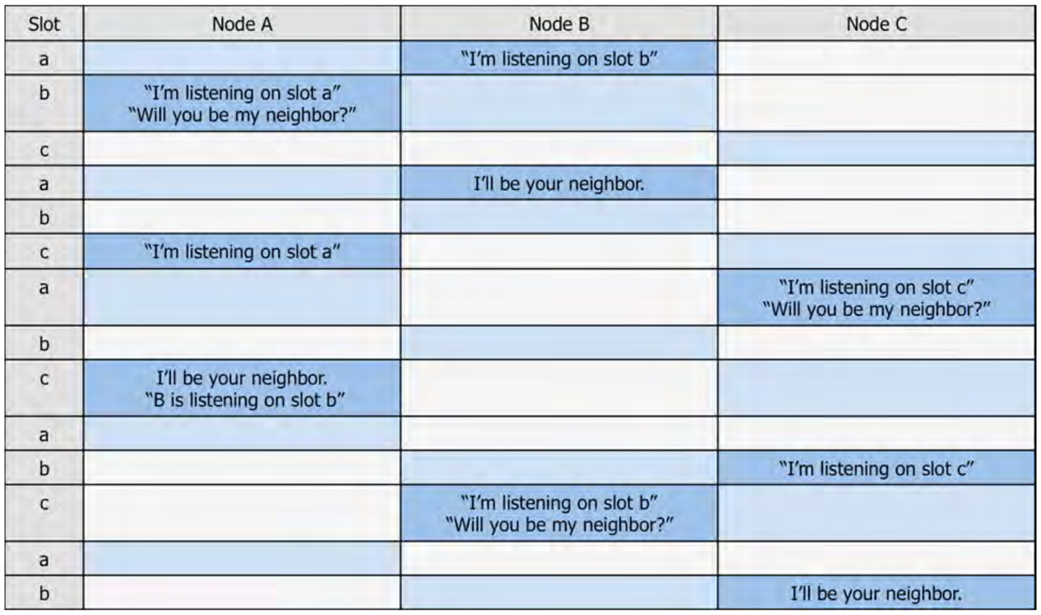
In the video-on state, which the ADAPT nodes entered after beginning to track an intrusion so that video or images could be used to capture the target and be transmitted back to the base-station for use in classifying a threat by a human in the loop, 1556 mW were consumed by each node. Nodes in this state also had all of their sensors running, their Wi-Fi enabled, and utilized their ground radio broadcast slots. They could operate in this state continuously for 3.2 days.

Finally, in the full power state, defined as the dual-core processor running at full capacity, all sensors and video on, and all radios being used to simultaneously transmit packets, the ADAPT nodes utilized 3156 mW of power. It would not be likely that the sensors would ever achieve this state, but this figure is included here in order to give perspective to the power consumption of the other operating states. Throughout the entire testing process with the nodes, they did not run in this hyper-vigilant state, but were instead in a testing/full mode where all radios were constantly enabled. This provided an operational lifetime of approximately 1.6 days [53].

3. ADAPT Sensor Node Operation

a. Communication Protocol

Communication between ADAPT nodes was conducted wirelessly via their onboard 900 MHz ground radios, utilizing the SAS MAC protocol [53]. This protocol utilized time division multiple access (TDMA) in combination with frequency division multiple access (FDMA) in order to conduct scheduled communications. This meant that the frequency spectrum of the SAS radios was divided into sub-frequency slots, and those slots were further subdivided into different time slots of 4.6 to 8.6 milliseconds in duration [53]. Nodes, at startup, would randomly select a slot to begin listening on. They would then scan the other slots and advertise their listening channel. Once a neighbor introduced them on their listening slot, they would jump to that neighbor’s listening slot and complete the three-way handshake. After this neighbor establishment has been completed, the node would then return to a state where they only monitor their slot, thus reducing the radio duty cycle of the nodes to less than one half of one percent [53], meaning nodes would use their radio less than one percent of the time, thus saving power and prolonging the nodes' life cycles. A depiction of the neighbor formation is shown in Figure 9.



Slot	Node A	Node B	Node C
a		"I'm listening on slot b"	
b	"I'm listening on slot a" "Will you be my neighbor?"		
c			
a		I'll be your neighbor.	
b			
c	"I'm listening on slot a"		
a			"I'm listening on slot c" "Will you be my neighbor?"
b			
c	I'll be your neighbor. "B is listening on slot b"		
a			
b			"I'm listening on slot c"
c		"I'm listening on slot b" "Will you be my neighbor?"	
a			
b			I'll be your neighbor.

Figure 9. Network Formation, from [53]

Data was transmitted in protocol data units (Figure 10) of variable length, called bundles. Each bundle could contain one or more packets of various types and lengths, but the bundle had a 230 byte limit [53].

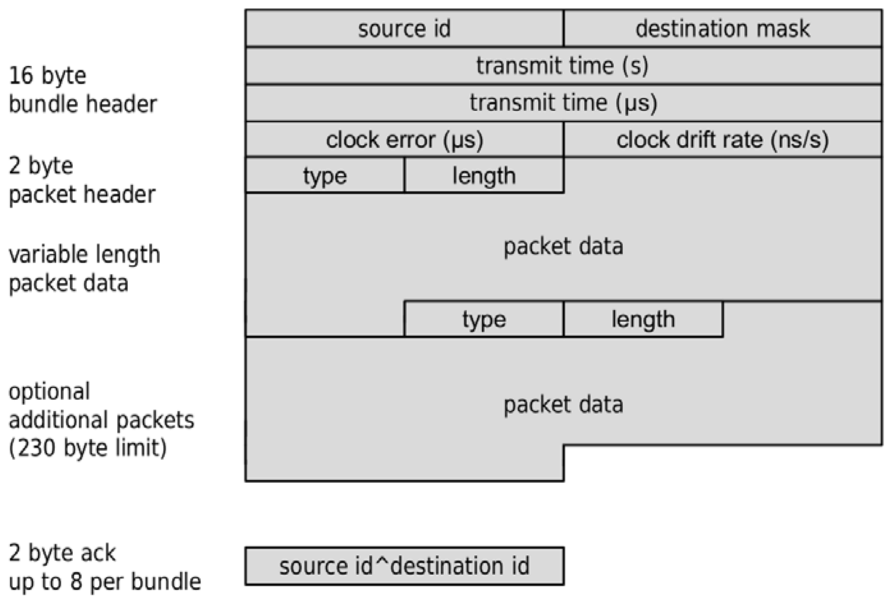


Figure 10. Bundle Format, the Data Protocol Unit for SAS, from [53]

b. Network Formation

The ADAPT nodes were capable of autonomously forming a WSN, making them suitable for use with MSAT since they would not need to be manually configured by warfighters employing them on the battlefield. To form this WSN, the nodes had to complete several tasks, to include neighbor discovery, time synchronization, and location determination.

To discover neighbors after being emplaced, a three-message handshake was used, shown in Figure 9. The neighbor node completed the handshake, by transmitting a final acknowledgment. The sharing of already-discovered nodes with new neighbors during the discovery process was designed to form the network such that an exponential growth of handshaking was not needed; while nodes would initially have few entries in their neighbor table, new neighbors would rapidly be discovered through neighbor-sharing [53]. Each node was limited by the SAS protocol to a maximum of eight neighbors.

Since the SAS protocol utilized TDMA and communication was scheduled, meaning that nodes received and transmitted only on a limited number of specific time slots in order to minimize power consumption by the radio, achieving time synchronization between nodes was vital. The initial time estimate per node was achieved through the GPS receiver on the node. This method of achieving an initial time fix was only partially reliable due to the known inaccuracy of the ADAPT GPS receiver onboard the node [53]. Therefore, an over-the-air (OTA) time synchronization algorithm via the ground radio protocol SAS was also utilized [53]. Due to inter-node clock drift, an algorithm using clock error and clock drift rate fields in the ground radio packet header enabled the measuring of inter-node clock drift and the resulting necessary time slot adjustments [53] to ensure the continuing ability to communicate between nodes. Every four hours, the nodes were programmed to obtain new time synchronization via GPS or OTA on ground radio [53].

As part of forming the network, nodes needed to obtain position fixes in order to report their locations to neighbors and any C2 base stations being monitored by human operators. To be useful, threat and detection data gathered by the nodes also depended on the nodes accurately obtaining positions. Nodes obtained a position fix in one of two ways: using the internal GPS receiver or estimation based on neighbor locations. Nodes with access to GPS satellites would obtain a fix upon being powered on and would remain in the GPS-on state for an hour, before transitioning to a less frequent GPS check. In a GPS-denied environment, (i.e., due to being underground, indoors, or jammed due to enemy activity) nodes obtained estimated locations by conducting a center-mass calculation using the positions of its neighbors [53].

c. Threat Detection and Tracking

Despite the variety of potential onboard sensors, the ADAPT nodes were adopted for use within MSAT at a stage in the prototyping process when the nodes relied exclusively on the PIR sensor for threat detection and tracking. This meant that there was no software yet developed for utilizing the seismic sensors and there were no functioning drivers to be able to utilize the cameras.

PIR sensors work by electronically sensing infrared light given off by an object moving through the sensor’s field of view. Any object, with a heat differential in respect to the surrounding ambient temperature, moving in front of the node’s PIR sensor would trigger a detection event. Upon a detection being triggered by the PIR sensor, the node would process the event—determining whether this detection was part of a sequence of detections from other nearby nodes (i.e., a track) and which neighbors to share the new detection with based on sharing-parameters set on the nodes. Nodes were set to send detection and tracking information with neighbors located within a 100 meter radius of the event.

Three sequential detections within the local vicinity (i.e., 100 meter radius) would prompt the creation of a track by the detecting node. A detection by a node with a track already existing in the local area would prompt the extension of the existing track: the bearing, speed, and location of which would be processed locally by the detecting node and shared with its neighbors.

Detection and track timeouts could be adjusted within the SIS database process on each node, so that nodes would drop detections and track records from their databases after a certain time period had elapsed. By default, detections were set to timeout after one minute and tracks after ten minutes.

d. Data Sharing

In the ADAPT WSN, nodes did not have a global knowledge of the network and only shared data with direct neighbors. This reduced the processing demands that would have been created should there have been the requirement of maintaining a large routing table and implementing a corresponding routing algorithm. In order to propagate information, nodes would only send information to neighbors that the neighbors defined as interesting. Interesting information was defined as detections within a 100-meter radius of the neighbor, or a track that had encroached into its area.

Each node maintained the locations of its neighbors, the local detections, and tracks in separate tables located in the SIS database process running on its operating system. New information received from neighboring nodes via the SAS ground radio protocol would be inserted into the appropriate table in the database. Each node also maintained tables of instructions from its neighbors regarding what kind of information each neighbor found interesting, so the node would know what information to send to its neighbors. In such a manner, information would propagate from one edge of the network to the other, one hop at a time. This precluded the need for a routing algorithm and end-to-end message addressing. It also resulted in redundant messaging, which provided network resilience in the face of possible jamming, destruction, or malfunction of individual nodes [53].

Additionally, nodes configured to utilize Wi-Fi could be programmed to send their SIS data (detections, tracks, and node locations) over an 802.11 link to the specified IP address of the base station, via port 10000 on the base station machine. This would allow for a human in the loop to monitor the entire sensor field. The operator could also issue instructions to the nodes via the SIS process, such as instructions to perform a simulated detonation by flashing the nodes' onboard lights, or to modify tables or data entries.

4. ADAPT Sensor Nodes Limitations

a. Inoperative Cameras

As previously mentioned, the software drivers that would have allowed for the use of the two cameras on board each node were not completed. Picture and video could be captured and stored locally on each node, via a workaround method, but this data could not be sent over the network to the base station for monitoring in real-time, making it of little use for surveillance and security.

b. Undeveloped Threat Tracking and Classification Algorithm

Software developed for the ADAPT nodes, at the time of MSAT's implementation, only allowed for intrusion detection based solely on PIR triggering and limited tracking based on sequences of detections. The nodes therefore had very limited ability to classify and analyze the detected intrusions into the sensor field [53]. Classification would therefore be limited to what could be provided by tracks, such as the estimated speed and bearing of the intruder, as well as the number of tracks created, which could possibly give an indication as to the number of intruders.

5. Testing the Adapt Sensor Nodes

a. Summary of Action

We traveled to Aberdeen Proving Grounds, 28 April–9 May 2014, to support testing and demonstration of the prototyped wireless sensor network being developed by the DARPA ADAPT program. All tests were conducted at the location of an old aerosol testing facility that was being utilized by Edgewood Chemical Biological Center (ECBC) staff. We worked closely on this project with several contractors working on the ADAPT program, as well as with the project manager for ADAPT. During our stay at Aberdeen, before the final demonstration to the Vice Chairman of the Joint Chiefs of Staff, Admiral James Winnefeld, we conducted many tests of the ADAPT nodes that yielded interesting results pertinent to the reliability and employment considerations of the system.

b. Testing

We conducted daily tests of the ADAPT WSN while at Aberdeen, to include four tests of over 60 nodes. All tests were organized in a similar manner. For each test, we hand-emplaced the nodes (shown in Figure 11) around the building, forming a sensor field capable of tracking an intruder walking around the building. The building was four-stories high and the wireless access point for the system was positioned on the building's roof. The terrain for the test consisted of slightly rolling hills to flat ground, with medium to high grass in the area immediately surrounding the building. Further away from the building (about 50 meters), the vegetation turned heavily wooded, with high trees and rich foliage. A flat, paved road ran straight north from the building, with grass along the edges of this road, giving way to densely vegetated woods 25 meters away from the road. Sensors were emplaced along both sides of the road to a distance about 200 meters up the road, away from the edge of the building. The testing site field measured approximately 800 meters in length oriented north to south, and 400 meters in width from east to west. Nodes were spaced, along a likely avenue of approach within this field, approximately 20 meters apart from each other and with varying density depending on the micro-terrain. The weather during testing ranged from heavy rain, to overcast, to sunny and clear; the wind ranged from heavy to calm. All testing was conducted during daylight hours. The testing site with a typical sensor node deployment, indicated by the blue triangles, is depicted in Figure 11.

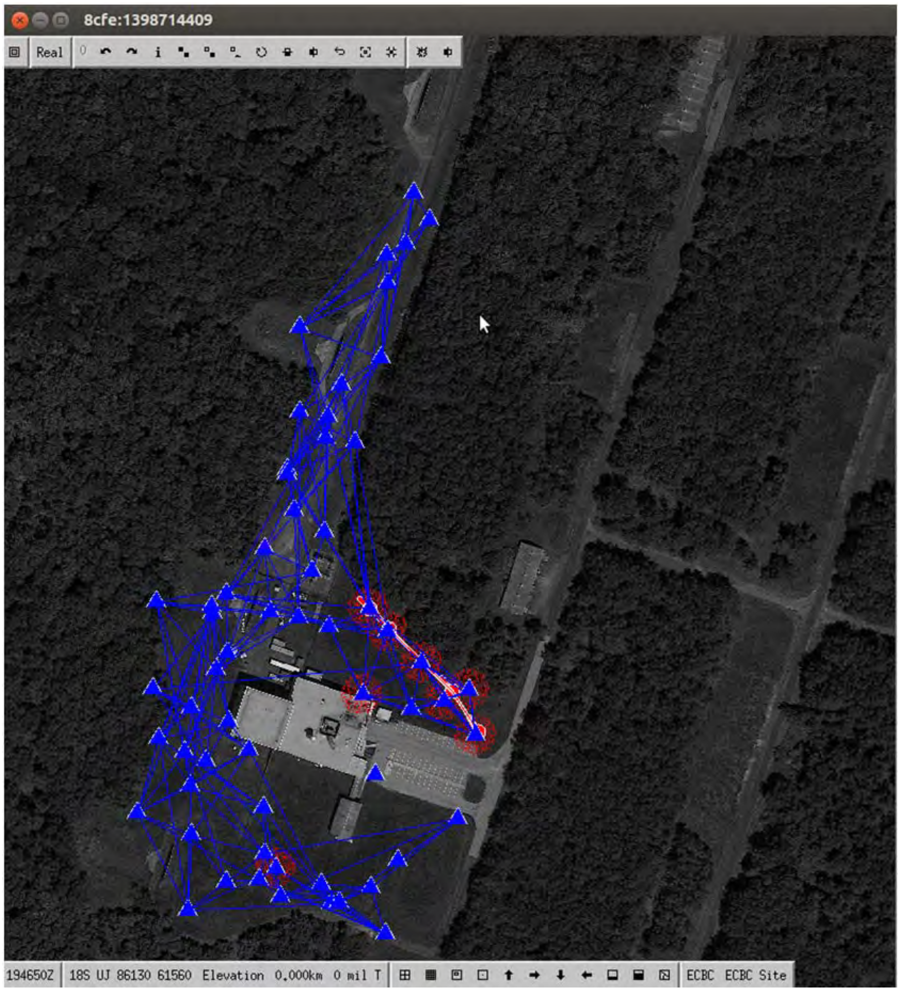


Figure 11. Aberdeen Test Network Formation with Neighbor Links

We emplaced the sensor field one time each day, but during that period ran several iterations of remotely bringing the network down and allowing it to autonomously reform. This capability to remotely bring the network up and down did not exist prior to testing in Aberdeen, but we developed a process that allowed for this capability using TCP over the 802.11 network. The ADAPT team also experimented with the transmission power of the SAS radios on board the nodes. Early in testing, we had tried turning the transmission power down to 15dBm, but later increased the transmission power to 23dBm as it yielded more consistent transmission of detections amongst the node’s neighbors.

We also ran several iterations of creating tracks through the sensor field each day. We took turns acting as intruders, creating tracks for the sensors as we walked around the building and along the northern road. For the first week, we tested only a single intruder at a time, creating a single track. During week two, we introduced a second intruder and tested the system’s ability to track two simultaneous targets. During this second phase, we attempted to keep the two tracks as discrete as possible so as not to confuse the system in the early stages of development, since the system’s tracking capability has not yet been refined.

We also tested the ability of the system to remotely kill an intruder as it moved through the sensor field. This was accomplished through a process that allowed the operator of the system to draw a radius around a geographical location, in which all nodes detonated if a track was created inside the circle. Turning on the nodes’ camera flash and LED light simulated the detonation of nodes—the nodes did not have a munitions payload or any means to detonate the payload.

As the nodes’ camera drivers did not function properly, rendering the cameras impractical for our use, two IP cameras were placed in the field to provide a real-time visual-monitoring capability of the sensor field, at the command center. One camera was placed looking north along the road and the other was placed south of the building. This provided the capability to see the intruder as it moved through the sensor field creating tracks. It also allowed the operator to visually observe the flashes of nodes as they were detonated in the simulated killings of intruders. This provided immediate feedback on the accuracy of the detonation.

As we tested each day, we made refinements to the positioning and orientation of the nodes based on the capabilities and limitations of the system as they were learned. Considerations for the positioning of the nodes included distance between neighbors, radio transmission power, and the orientation of the PIR sensors in relation to the terrain, vegetation, and neighboring nodes. The sensor network operator also adjusted the timing and radius of the detonation for the simulated munitions as testing progressed.

On the final day of testing, we conducted a demonstration for VIPs, which consisted of executing the events described above. First, the network was allowed to autonomously form from a down state, with no links previously existing between nodes. Observers witnessed the time that it took for the entire network of nodes to obtain GPS locations, begin reporting to the sink node (i.e., COC), discover neighbors and form links. Next, two intruders walked through the field simultaneously, creating separate tracks. Finally, the WSN operator in the COC attempted to "kill" one of the intruders at two separate points along the route, each observable by a camera.

c. Results

Throughout testing, we observed that the autonomous formation of the wireless mesh network was quick and reliable. On average, it took between 60 and 70 seconds for the first node to report to the COC after starting the network, 90 seconds for the first links between neighbors to be created, and four minutes for the entire network to be formed. On two occasions, ten nodes were purposely left in the off state and brought up after the formation of the network. These nodes were able to autonomously form neighbor connections and integrate into the network with limited errors. Physically emplacing nodes was a non-trivial task, and it took an average of 30 minutes for three people to emplace 63 nodes. This was equivalent to a rate of .7 nodes per minute to be emplaced for a single person.

All testing was done with the nodes in a testing/full state at all times—vigilant power mode was never utilized, as it had been too unreliable. Also, all nodes were directly reporting via 802.11, requiring constant use of the Wi-Fi antenna. This meant that due to high power consumption rates, life cycle times for the nodes was a maximum of two days, rather than the greater than 20 days that is desired by the ADAPT team, in fulfillment of requirements of the client.

At the lower transmission power of 15dBm for the SAS radios, we observed that nodes in the southern portion of the sensor field had difficulty forming neighbor links.

This led to unreliable tracks that were often not continuous because of the difficultly of sharing detections between neighbors. For this reason, even though tracks were still recognizable as single tracks to the human operator monitoring the system graphically, single tracks often registered in the system as multiple tracks. By turning the transmission power up to 23dBm, we observed better neighbor connections between nodes and more continuous and reliable tracks.

The environment had a major effect on the performance of the nodes. Heavy rains that persisted for three entire days of testing subjected the nodes to flooding, and upon inspection we noted that the inside of some nodes had become damp after remaining out in the rain all day, with some being partially submerged. The rain also affected the reliability of the PIR sensors, simultaneously degrading the ability of the system to form tracks and increasing false detections. As one would expect with most PIR sensors, we also experienced the same reliability issues during periods of bright sunlight. The best performance was achieved during overcast conditions. High winds caused a proliferation of false detections, with the nodes generally performing better under calm wind conditions. Vegetation also was a problem—high grass and foliage swaying in the wind tripped PIR sensors. After maintenance workers at the testing facility cut the grass around the building, we noticed far less false detections. We found the reliability of the PIR to be heavily dependent on the environment; implementing a more refined detection system would require the use of multiple sensor modalities.

Testing the network with two simultaneous intruders walking through the field yielded mixed results. We discovered that the two tracks had to be very distinct, meaning separated significantly by time and space. To get two clean tracks, we had to start the intruders at opposite ends of the field, and they could not cross paths at the same point in time, otherwise the tracks would become confused.

The effectiveness of detonating nodes in order to kill intruders increased as the operator became more experienced with his timing, so this process requires a highly involved human in the loop.

Figure 12 is a screen shot from the COC that depicts the nodes (blue triangles) and a red track generated by a single intruder. In this test, the track is broken/interrupted as the intruder walked around the corner of the building and the nodes could not associate the events from the previous detections and tracks, to the current ones being reported. This does not have a direct impact on the operator's ability to determine that an intrusion is occurring, but does introduce ambiguity as to how many intruders are present and their direction. If the operator is vigilantly watching the scenario unfold on a device, then it would be apparent, as the track stopped at a certain point and continued later on, that the tracking of the intruder was broken. Figure 13 displays an unbroken track following a single target, while Figure 14 displays the tracks of two objects that started in different areas of the sensor field.

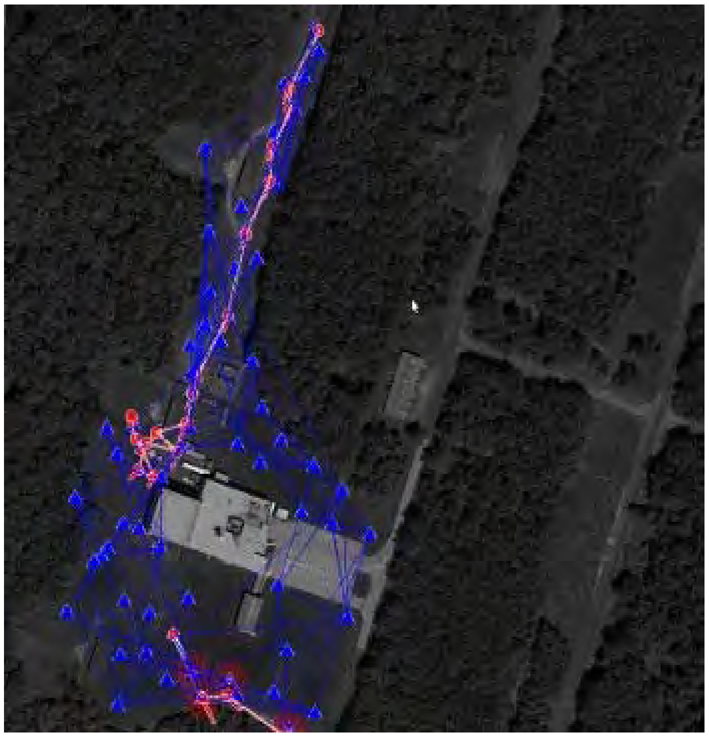


Figure 12. Broken Track Following One Person

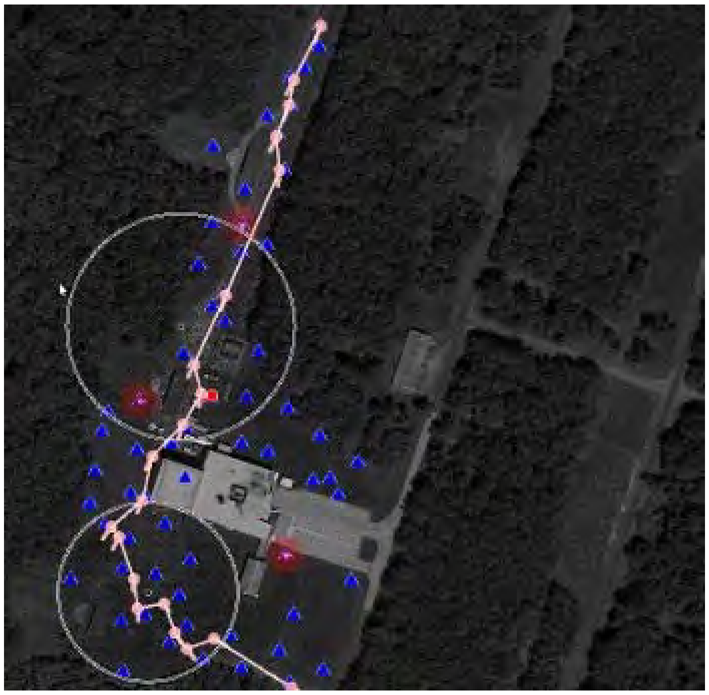


Figure 13. Continuous Track Following One Person

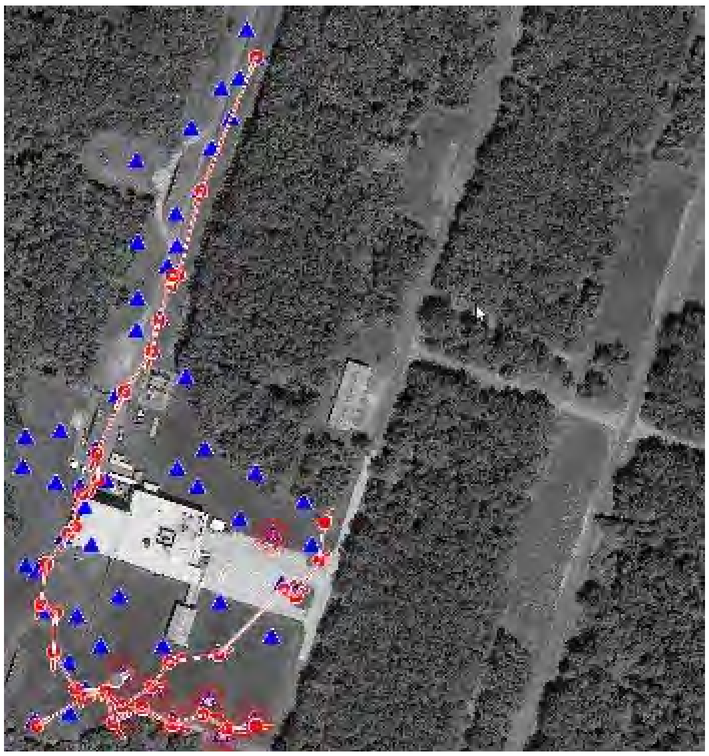


Figure 14. Continuous Track Following Two People

d. Conclusion

The performance of the ADAPT sensor nodes during field testing validated the choice for their incorporation into MSAT, but also revealed shortfalls that would have implications for their inclusion in the automated surveillance system. The sensor nodes generally met the requirements for use with MSAT: the nodes were low-cost, wireless, TCP/IP compatible, performed network formation autonomously, and executed basic detection and tracking algorithms. However, the shortfalls of the nodes would present some difficulties. First, the nodes relied solely on the PIR sensors for all threat detection and tracking. This single sensing modality limited the likelihood of conducting any advanced threat classification with the system. Second, the reliability of these PIR sensors was highly dependent on environmental factors such as vegetation, wind, rainfall, and lighting conditions, leading to unpredictable behavior and a high incidence of false positive detections. Unpredictable sensor behavior would have the potential to confuse warfighters operating MSAT. The effect of this shortcoming on the actual implementation of the MSAT prototype would be a system that forces the warfighter to rely more heavily on manually classifying intrusions reported by the sensor system through the use of video. This reliance on video for manual threat classification would be further exacerbated by the existence of only a single functioning sensing modality on the sensor nodes, ruling out the potential for automating threat classification by using multiple sensors in collaboration.

Having established the importance of video, the next obvious problem that would have to be overcome was the lack of functioning cameras on the sensor nodes. To solve this issue, the MSAT reference design would have to incorporate separate cameras into the system in order to provide video feed for the warfighter, at least until functioning drivers were created that would allow for use of the internal cameras. The effect of this modality is to rely on the ADAPT sensor field as a tripwire system calling the attention of the individual monitoring the field of interest to specific locations in that field.

B. APPLICATION SOFTWARE

We designed an application to act as the interface between the sensor nodes and the warfighter using COTS mobile devices (Figure 15). In order to achieve our goal of interoperability with the widest variety of device models, and in keeping with the bring your own device (BYOD) philosophy of a military that has not yet decided on using a universal standard, we created an HTML5-based application utilizing the client-server model. The building of our application was thus logically divided into two functional areas—the application server that would interface directly with the nodes and serve content to clients, and the client-side code that would execute on the users' devices.

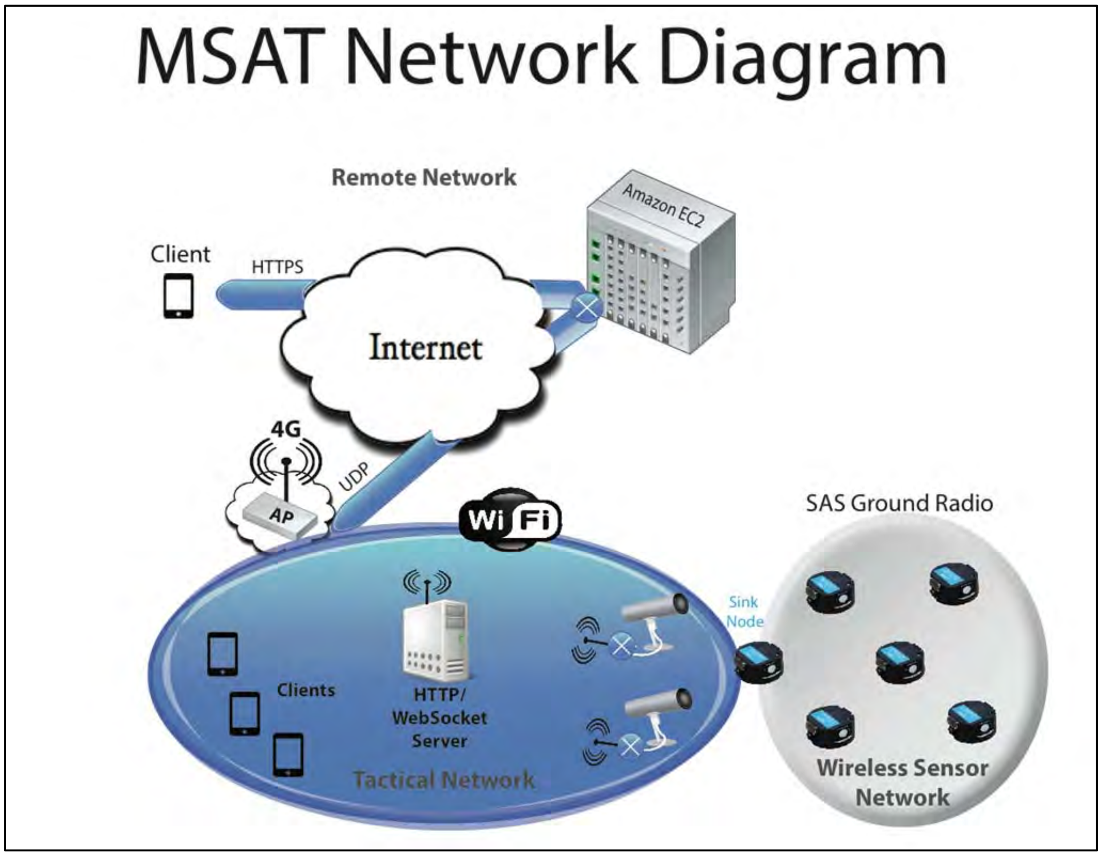


Figure 15. MSAT Physical Topology Diagram

1. Server

MSAT's application server was built to accomplish several tasks. First, it would act as a traditional HTTP server, serving static content to clients. Second, the server would interface with the sensor nodes, receiving data on node locations, intrusions, and targets as they were tracked through the field in real time. Third, the server would process this data and serve it to connected clients in a human readable format through the user interface. Fourth, the application server would facilitate tracking friendly positions, the sending of messages and reports on the network, and monitoring the sensor video feed. All of this functionality would be delivered for the purpose of maximizing situational awareness for the warfighter utilizing the system.

a. Hardware

To run our application server, we utilized a Lenovo ThinkPad T510 with a dual 2.67 GHz Intel Core i7 CPU and 4 GB RAM running a 64-bit version of the Ubuntu 12.04.4 LTS operating system. This laptop contained a Centrino Advanced-N + WiMAX 6250 Wi-Fi adapter by Intel corporation, compatible with 802.11 a/b/g/n, and a 10.8 volt Sanyo model 42T4791 6-cell lithium ion battery with 47520 mWh capacity. This laptop was selected for its compatibility with the Ubuntu operating system, which was free and open-source, making it friendly for use as a development environment for prototyping.

b. Version 1: LAMP Stack

We initially attempted to build our application server utilizing a Linux, Apache, MySQL, PHP (LAMP) stack because of its open-source nature, widespread use on the Internet, and thorough documentation. The LAMP server was installed on the application server laptop. This server would provide connected clients with dynamic, real-time updates through Asynchronous JavaScript and XML (AJAX) requests. To service AJAX requests from clients, a PHP script on the server queried for node information stored in the MySQL database. The MySQL database contained three tables: locations, detections, and tracks. In order to communicate with the sensor field and populate the tables on the server with the relevant information in real-time, we built an interface between the server and the sensor nodes. Figure 3 depicts this implementation.

(1) Server-Sensor Field Interface

We ran a SIS database instance on the application server machine and configured the sensor nodes to send their tables to the IP address of the application server on Port 10000 over 802.11. A Bash script was written which invoked an infinite loop that queried the SIS database, writing the results of the query into a text file on the server. After writing the results of the query to a text file, the loop invoked a Python program to parse the data contained in the text file into the MySQL database, utilizing the MySQLdb library as the interface between Python and MySQL. Finally, the script cycled to the start of the loop to repeat the process. This kept the MySQL tables up to date with node information provided by SIS.

(2) Data-Transfer between Client-Server

The primary purpose of the application server was keeping clients updated with the latest state of the sensor field through the User Interface (UI) on the client mobile devices. The state of the sensor field consisted primarily of node locations, intrusion detections, and tracks.

The UI was designed as a single page application (SPA), meaning that clients were served the static HTML5 and JavaScript upon initially connecting to the Apache2 HTTP server. The static content included the layout of the page and styling in the form of HTML, CSS3, map imagery of the local area, and JavaScript code that contained instructions for the client's browser to execute. The dynamic content would be updated without having to reload the webpage.

Through the execution of the JavaScript instructions in the client browser, the SPA dynamically updated changing node locations, detections and tracks as they were processed by the server and forwarded according to the JavaScript programming. This was accomplished through implementing AJAX polling to retrieve the updates from the server. Specifically, HTTP requests for sensor field updates were made to the Apache2 server at a rate of every second using the native JavaScript function *setInterval*, which executed a function at specified intervals, and the *XMLHttpRequest* object, which allowed for the creation of HTTP requests to the server originating from the client's browser, without having to reload the client's webpage.

On the server, a PHP script was written to handle the HTTP requests from the clients. When invoked by the client through the Apache2 server, the PHP script queried the MySQL database that contained the current state data for the sensor field. The script then parsed the results of the query into JavaScript Object Notation (JSON) and returned the JSON object to the requesting client via the HTTP server. The JavaScript on the client would then parse this information into a human-readable format generating the display of node locations, tracks, and detections as graphics on the map overlay. As previously mentioned, this process was executed every second for each client connected to the application server through a persistent HTTP connection.

The implementation of a LAMP server allowed for the building of an initial working prototype for MSAT, but performance limitations quickly became obvious. During testing while developing the application, the application suffered from severe latency, which resulted in a slow, unresponsive UI on connected client devices. It was noticed that this problem grew noticeably worse as more clients simultaneously connected to the application server. The cause for this latency was predicted to be the Apache2 server's multi-threaded nature and the blocking characteristic of querying the MySQL database. Not only did every client's request for static content generate a new Apache2 thread, but each client generated HTTP requests at a rate of one per second while connected and each request spawned yet another thread by the Apache2 process running on the server. Additionally, each AJAX polling request invoked a new MySQL database query through the PHP script. Since the database queries were blocking, meaning that they were executed synchronously such that only one query could be executed at a time and all other threads had to wait in a queue to be serviced, the database querying manifested as a choke point on the server. This choke point slowed responsiveness.

Additionally, the LAMP-based application server lacked any video capability, as the cameras on the ADAPT nodes were not accessible due to non-functioning drivers. Simply meeting the demand of serving node data to connected clients stressed the server beyond being usable, so the streaming of video to clients was not considered feasible without a major design change on the server.

c. Version 2: Node.js Server

To solve the performance issues of the LAMP server that rendered the application unusable, a complete redesign of the server was conducted utilizing different server-software. Node.js was adopted as the solution due to its asynchronous model of execution and its ability to create fine-grained networking applications through the JavaScript programming language. Figure 16 is a diagram depicting how this was implemented.

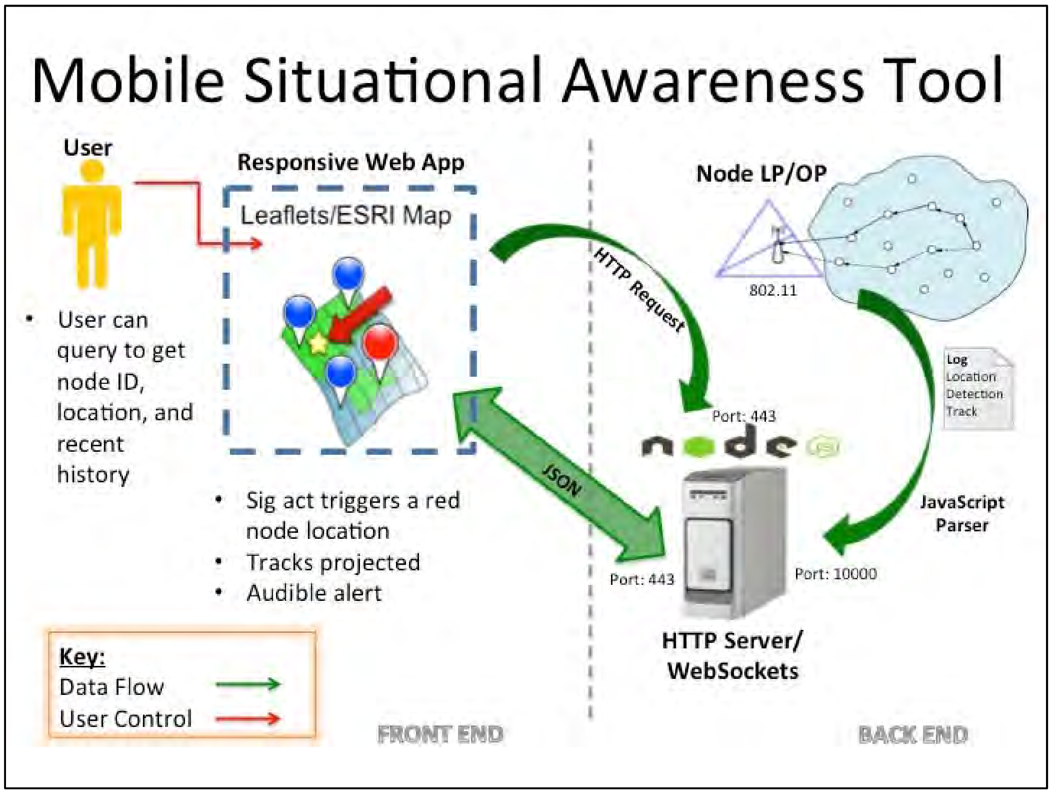


Figure 16. MSAT Logical Topology Diagram

d. Performance Test: Comparison of Apache2 versus Node.js Implementations

To determine which paradigm would provide the best performance, Apache2 and Node.js were compared in a benchmark test. The results of this test were used to decide whether or not to utilize Node.js over Apache server, which had already displayed disappointing performance.

The overall goal of the test was to collect quantitative data about the performance of the two servers designs under various loads. This was an important step in understanding the capabilities and limitations of each technology and would aide in choosing the correct server model with regards to the web application portion of the overall system. Two different test scenarios were conducted for each server setup. The first scenario tested the each server's ability to serve a static web page to a large number of non-persistent client connections. The second scenario tested each server's ability to service a persistent connection where a large number of clients were simultaneously connected and continuously making HTTP requests of the server. The later scenario simulated the use of AJAX for creating dynamically updated webpages.

To conduct the test, we decided that it was more important to utilize a sterile direct, end-to-end connection between the simulation computer and the server rather than a more complex network. During initial trials, it was recognized that testing on a wireless network introduced uncontrolled variables, where results varied and were not replicable. Some of the problems in attempting to use a wireless network to precisely perform server load testing included the higher frame error rate due to the wireless medium and interference from nearby stations. We also attempted to conduct the test over the Naval Postgraduate School intra-network with the server connected by wire. However, we noticed early in the trials that the results were skewed by current network conditions such as traffic intensity. This intensity would ebb and flow throughout the day based on usage. This made recreating the network conditions of each test nearly impossible. In an effort to minimize the uncontrolled variables, we decided to create a simple network, utilizing an Ethernet connection between the load simulator and the server. This minimized the number of variables in the experiment, in order to allow for a strictly controlled test of the servers.

We ran our server on a desktop computer with a 2.66 GHz Intel Core Duo processor and 2 GB of RAM. We used 64bit Ubuntu 12.04 LTS for the operating system. On our server machine, we installed two different server architectures. The first was Apache2 and was configured to listen on port 3000. We made no special modifications to the standard software. We installed our Node.js server on the same machine. Our Node.js server was custom programmed to serve our UGS application, in an asynchronous manner. This was also configured to listen on port 3000. Since both servers were configured to listen on the same TCP port number, only one server could be run at a time.

After looking at various load simulation tools, we decided to use the Java-based *JMeter*, an open-source load generating software, to generate client requests. We also decided to use the more powerful computer to simulate the client requests, because we anticipated that generating so many concurrent client threads would be a computationally intensive task.

In order to setup the simple network, depicted in Figure 17, we statically assigned IP addresses to each computer's Ethernet interface. This was relatively simple to do using standard Linux networking commands via the command line interface in Ubuntu. Also worth noting, we were able to use a standard Ethernet cable rather than a crossover cable, as most modern drivers are able to recognize such an Ethernet connection directly between two computers.

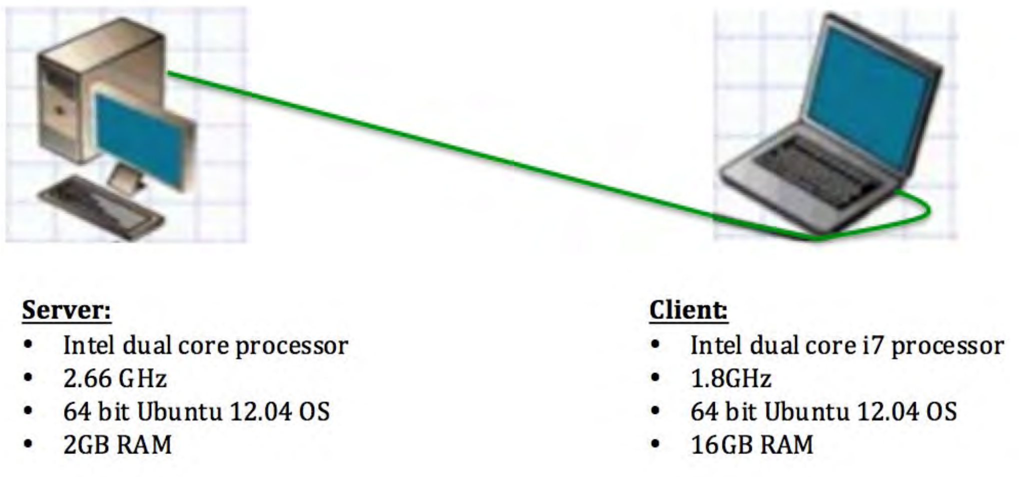


Figure 17. Testing Network Configuration

The first test conducted was the standard load test. This simulated a large instantaneous burst of client requests. All client requests were generated simultaneously, with no ramp up time between the requests. Worth mentioning is that while we simulated thousands of simultaneous client requests, in actuality, since there was only a single computer generating the requests, there was a small delay between these requests, so they were not truly simultaneous. During this test, we served a static HTML document through non-persistent TCP connections. This meant that a single client thread was only responsible for generating an HTTP request once per test. The clients' requests arrived in the server's queue nearly simultaneously. Once the entire queue had been serviced, then the test would be complete. This test was chosen to simulate the condition where a large burst of traffic suddenly arrives at the queue, which is traditionally one of the most stressful demands that can be placed on a server. This scenario was also useful for highlighting the potential strengths and weaknesses of different kinds of servers. During this test, we incrementally increased the number of simultaneous client requests, in thousand client increments starting from one thousand requests all the way up to twenty thousand requests. We did this first for our Node.js server and then for our Apache2 server. The results for response times are presented in Figures 18–21.

The second test was designed to more closely simulate the operation of the MSAT application, which required multiple *persistent* client connections. In this scenario, each client made multiple HTTP requests per second, in order to simulate achieving a real-time monitoring capability for the UGS system. In order to accomplish this, clients established persistent TCP connections, which stressed the server in a different manner than the first test. After the initial influx of client connections, the server had to constantly work to service pipelined client requests.

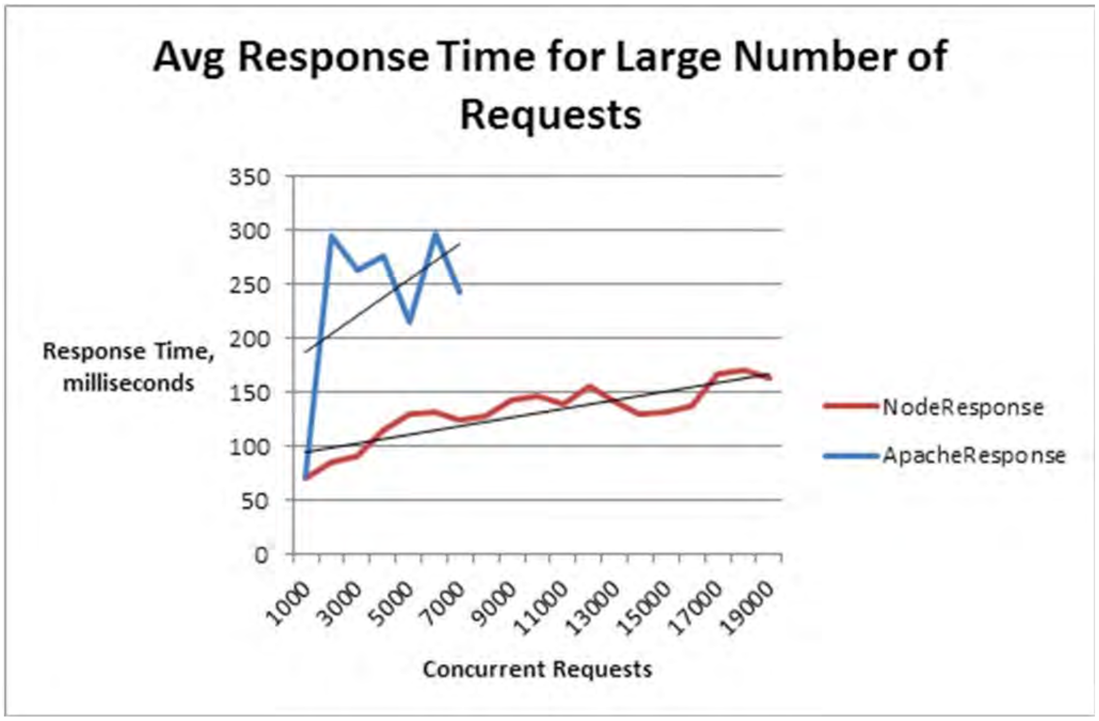


Figure 18. Comparison of Average Response Times for Concurrent Requests

As is apparent from the graph in Figure 18, at one thousand requests both servers had comparable response times. After this point, the Apache2 server began to slow down considerably, while the Node.js server gradually increased its response time. After the seven thousandth concurrent request mark, the Apache2 server was no longer able to service requests and approached the 100% error rate. The Node.js server easily surpassed the Apache2 server break point and went on to nearly double the allusive (i.e., C10k problem) ten thousand concurrent client connection mark for a web server. Past the nineteen thousand-request mark, the performance of the Node.js server quickly degraded as error rates grew unbounded.

In addition to utilizing *JMeter*, we wrote server side code that measured the service time of each request and calculated an average service rate per iteration of the test. Shown in Figure 19 are the service rates for the Node.js server.

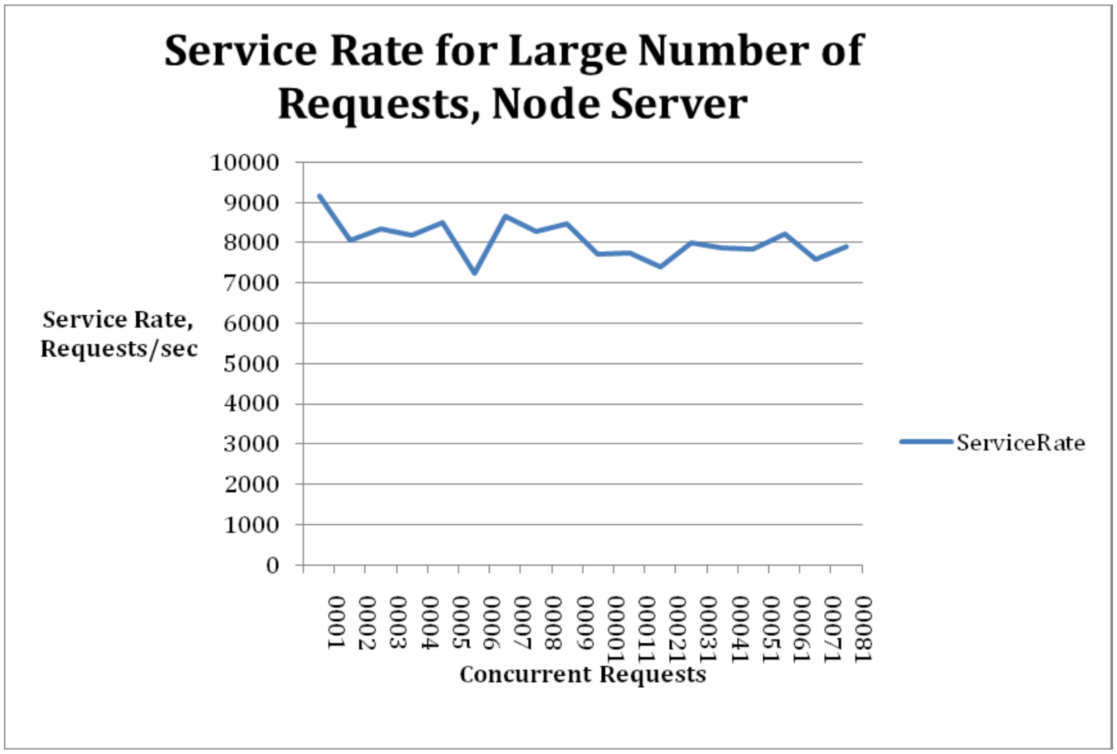


Figure 19. Node.js Server's Service Rate for Requests

The results were inconclusive and possibly indicated a slight decline in service rates as concurrent requests grow. This was unexpected, as we anticipated a more definitive decline in service rates as the number of requests grew. This may indicate that there was a bottleneck existing somewhere within the operating system, or at some location outside of our server logic. This could be further explained by the way the operating system deals with a large number of TCP connections.

For the second test, we further tasked each client to repeatedly make pipelined HTTP requests to the server, until one hundred thousand total client requests had been serviced. We started at one hundred simultaneous connections and incremented by one hundred clients, until one thousand concurrent connections were established. After observing the results from test one, we expected Node.js to greatly outperform Apache2, but the results shown in Figures 20 and 21 were surprising.

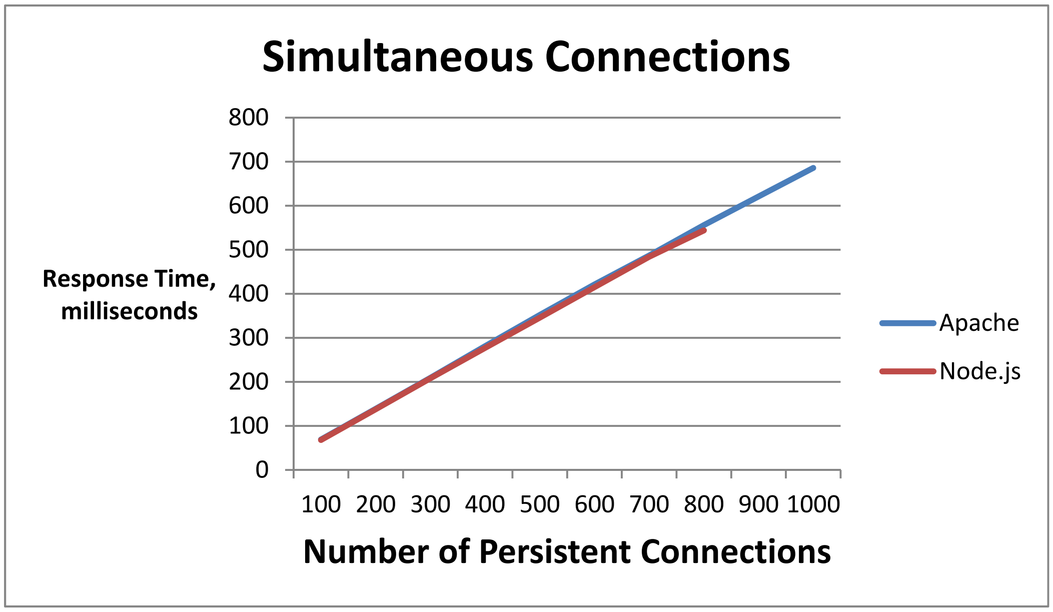


Figure 20. Comparison of Average Response Times for Persistent Connections

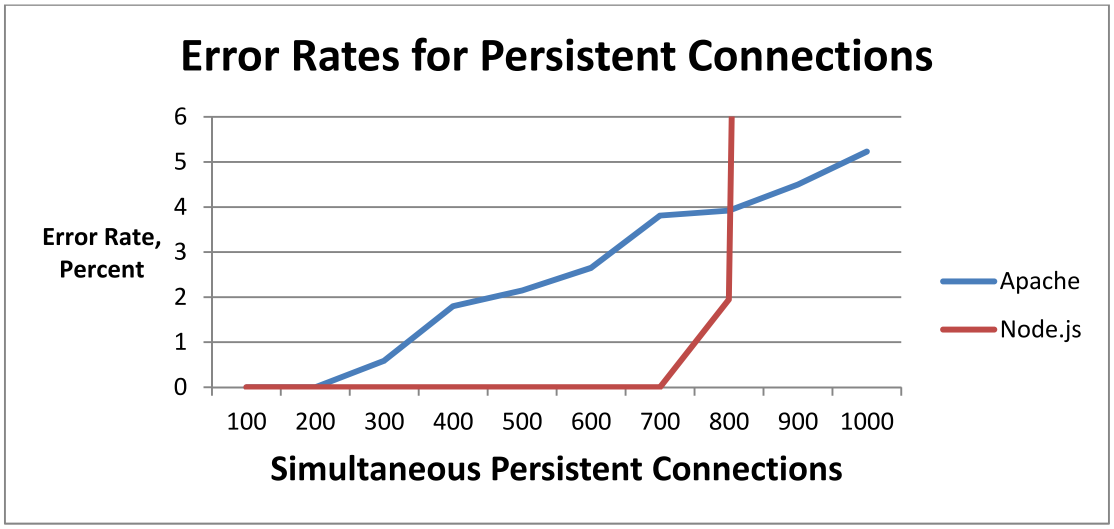


Figure 21. Comparison of Error Rates for Persistent Connections

In Figure 20, the Apache2 and Node.js servers performed almost identically, with Node.js having a slight advantage over Apache2 until the 800 connections mark. At this point Node.js suddenly failed and Apache2 continued on with error rates growing linearly. This is where Figure 21 is of interest, because Node.js experienced a zero percent error rate all the way up to the 800 simultaneous connections point, while the Apache server began experiencing errors as early as 300 connections. This indicates that there was probably an acute reason for the sudden failure of the Node.js server, and we speculate that this reason could be operating system limitations, or even limitations to the underlying configuration of our Node.js installation. We also have no definitive explanation for the gradually increasing error rate of the Apache server. This may warrant future exploration.

This testing led to interesting conclusions. We believe that the results of our second test validated our design decision to migrate our UGS application server from the multi-threaded model of Apache to the asynchronous Node.js. The reliability of the Node.js server all the way up to the 800 connection mark was encouraging and we believe made it a better choice for our application than Apache, as the need for more than 800 simultaneous users of our system is unlikely. In other words, we placed an emphasis on reliability for a reasonable number of simultaneous connections over the capability to handle an extreme number of connected clients but with a diminished overall reliability, even at lower numbers of client connections. The results of test one further validated this design choice because Node.js clearly was more capable of handling massive bursts of requests without errors.

Overall, due to a number of interesting research papers that have already addressed the performance of Node.js, we expected Node.js to excel during these two tests. However, with both tests, Node.js experienced dramatic failure points, rather than gradual degradations of performance. This would seemingly indicate that we did not approach the actual limitations of the Node.js server, but rather witnessed a potential misconfiguration or bottleneck that was capable of being overcome. This is in contrast to the Apache2 server that experienced gradually increasing error rates in the face of growing concurrent connections, which we believed was an indicator that the server model itself was being stressed to its eventual breaking point. For future work, we propose examining the potential sources of the Node.js failures in an effort to correct them. This would involve carefully delving into the configuration of the operating system and Node.js source code. After doing so, another benchmarking test should be performed.

Nonetheless, based on the results of this test, Node.js was selected for use to replace the Apache2 server used in Version 1 of the application.

e. Interface with Nodes

In Version 2 of the MSAT application server, the MySQL database was eliminated in an attempt to improve performance by fixing the bottleneck that had existed previously, when numerous blocking queries were executed by the AJAX version. This was enabled by the ability of Node.js to make direct queries to the SIS database process running on the ADAPT server through the *shelljs* module. Recall that the SIS process provided the interface with the ADAPT nodes, which sent pertinent environmental data to the SIS process running on the machine via port 10000. The *shelljs* module is an official Node library that allowed the execution of bash shell commands from within a Node.js application. The querying was executed in a non-blocking fashion through *shelljs*' ability to execute shell commands asynchronously. A callback function provided to the asynchronous *shelljs* query to SIS parsed the results for the query into a JSON object directly, thus eliminating the need for the parser written in Python on Version 1 of the server. This JSON was then transferred to the clients.

Figure 22 is a container diagram, highlighting the interactions between code modules that occur within the MSAT system. The two servers are depicted with their functions, as well as the communication that occurs between them, enabling the sharing of sensor field messages to the user's mobile device. In addition, the peripheral devices, IP camera, and nodes are shown and how they interface with the tactical network server.

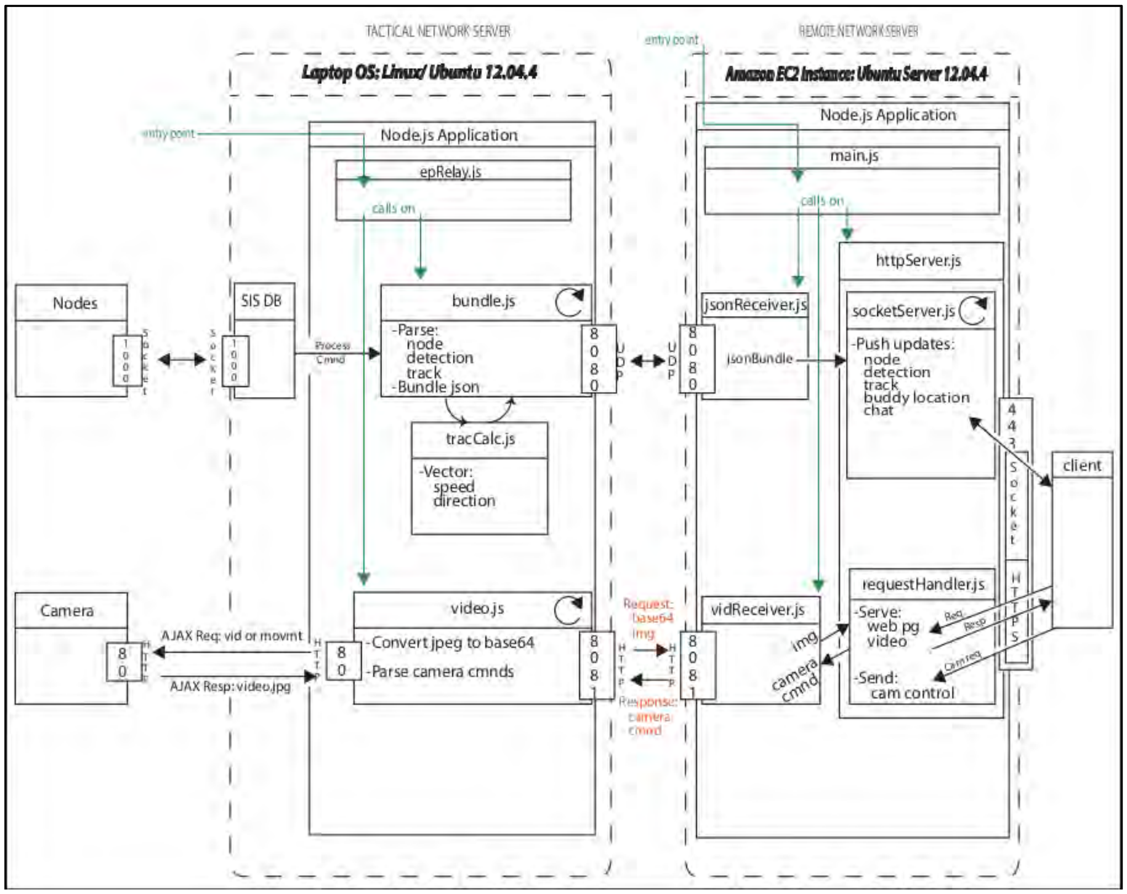


Figure 22. Application Container Diagram

f. *Replacing AJAX with WebSockets*

Server-polling through AJAX put unnecessary stress on the application server due to the inefficient generation of HTTP requests. With this model, clients requesting situation updates sent an HTTP request every second while connected, even when the server had no new information to push. To eliminate this unnecessary generation of network traffic and reduce total network latency, AJAX was replaced by the use of WebSockets. WebSockets allowed for the establishment of continuous, bi-directional connections between client and server entities. This provided the ideal instrument for real-time updates from the server because with WebSockets, data only had to be pushed over the network when there was new information to report, eliminating the unnecessary polling messages.

The use of WebSockets was enabled by the *Socket.io* module for Node.js. JavaScript on the client pointed to the location of the WebSocket (i.e., port number and IP address) on the server and established the connection upon the client initially being served the SPA. On the server, the *setInterval* method was used to query SIS every second, parse the result into a JSON object, and pipe this result through all open WebSockets to clients. This provided the real-time view into the sensor field.

g. Blue Force Situational Awareness: Locations and Chat

MSAT was intended to provide advanced situational awareness to users of the application, so besides just updating sensor field activity, the application provided the ability to track and communicate with friendly forces (i.e., blue forces) on the battlefield. This was accomplished through two means: plotting friendly locations and chat functionality.

The Geolocation API provided by HTML5, in the client SPA, obtained the user's location either through a Wi-Fi location estimation, which was accurate only to 500 meters, or through GPS when available on the client device. Each time the Geolocation API obtained a fix for a user's location, a callback function would be triggered that would send the user's coordinates via the WebSocket to the server. Every time the server received an updated location from a connected client, a callback would be triggered that updated the user location in an array used to track the locations of all connected users, and the updated coordinates would be sent out to all connected clients through their WebSockets, ensuring that each client maintained an accurate picture of friendly positions.

Through the SPA, clients had the ability to chat with all other users of the application, and the server provided the backbone of this functionality through WebSockets. Client's had the option of sending either private messages meant to be received only by a single user or public messages meant to be broadcast to the entire field of users. The server received messages generated by the client through the client's established WebSocket; upon receipt, a callback function on the server multiplexed the message appropriately. To be able to send private messages selectively to individual clients, an array was maintained on the server that associated WebSocket identification numbers (i.e., unique numbers assigned to each WebSocket by the *Socket.io* API) with client-chosen usernames in order to resolve the WebSocket identification number of the intended recipient. This WebSocket identification number therefore acted essentially as a client's address for private messages and facilitated its appropriate routing.

h. Remote Command and Control (C2) Configuration

The capability to access the MSAT application remotely (i.e., from outside the local 802.11 network) was sought in order to enable personnel located offsite to monitor the sensor field. This capability would support the use case of leadership located in a COC being able to see the same view of the sensor field as the warfighter, who would be using the application co-located (i.e., connected to the same local area network) with the sensors on the battlefield. It would also support the use case of monitoring the sensor field while no friendly forces were co-located with the sensors (i.e., completely remote monitoring).

To achieve this remote COC functionality in Version 2 of the MSAT application, the application server software was divided into two separate processes: a relay designed to be run on a machine located within the local 802.11 network, and an application server that could either be co-located with the relay running on the same machine, or alternatively run on a separate machine on a network outside the local tactical Wi-Fi network. Both processes were implemented with Node.js and written in JavaScript.

The purpose of the relay was to route data between the ADAPT sensor nodes and the application server. The interface with the ADAPT nodes was achieved through the technique detailed in the above section. Upon parsing the node data into a JSON object whenever it received results from a query to the SIS database, the relay would send the JSON object via user datagram protocol (UDP) to a server running on the remote server process located outside the local network. UDP was chosen as the protocol for JSON transfers due to its speed. The unreliability of datagram receipt with this protocol was deemed acceptable in this case because of how often new JSON objects were forwarded. The relay also fed camera images from the AXIS IP camera, which was integrated into

MSAT to provide a video capability, to the application server through an HTTP server listening on the relay server. The relay server also passed on camera control commands (i.e., tilting, panning, and zooming), originating from the user on the mobile device, from the application server to the HTTP server on the AXIS camera in the form of HTTP requests.

In the remote configuration, the application server was accessible by the relay through its public IP address. In our implementation, the remote server was installed on an Amazon EC2 instance running the Ubuntu 12.04 Server operating system. The relay, however, was located behind the network address translation (NAT) functionality provided by the Verizon MiFi (responsible for providing the Wi-Fi bubble locally), and thus was not directly accessible from outside the network. Therefore, the relay had to initiate all connections to the remote server due to the inability of the remote server to locate the relay behind the NAT of the MiFi.

The application server could be run on the same machine as the relay to give local users access to MSAT, it could be run separate from the relay to enable remote monitoring, or both on the local machine and remotely to allow both local operators and offsite COCs to use the application at the same time.

i. Security

After the remote COC configuration was introduced, Transport Layer Security (TLS) was implemented on the application in order to encrypt communications between clients and server.

The native https module in Node.js was utilized to create an HTTPS server that established the encrypted tunnels between clients the server. OpenSSL, an open source encryption-program for GNU Linux, was used to generate the private/public key pair necessary for using TLS. OpenSSL was also used to generate the certificate required for authenticating the server for the HTTPS protocol. The certificate was self-signed.

Basic authentication was programmed into the server through the *http-auth* library. The basic authentication protocol worked by maintaining a list of authorized users in a file named *users.htpasswd* on the server. This file contained a list of usernames and their corresponding passwords digests, which were the result of hashing the users' passwords with the MD5 algorithm [55]. Once a user requested the SPA from the MSAT server, the user would be prompted to enter a valid username and password. The server would then hash the password with MD5, and perform a lookup of this username with the password digest in users.htpasswd. If the pair existed, then the server would serve the SPA to the client who could begin viewing the sensor field. If not, the client would not be able to access the SPA.

The application's WebSockets were also encrypted via the WSS protocol, which implemented the WebSocket protocol over TLS. This functionality was programmed using the *Socket.io* module for Node.js.

j. Camera Solution and Streaming Video

Because of the limited ability of the ADAPT nodes (i.e., PIR sensors only) to conduct threat classification and analysis, another means was sought in order to satisfy the requirement that an automated LP/OP would have to fulfill the information requirements of the SPOTREP. With functioning camera drivers on the ADAPT nodes, video imagery could have been inspected by the human in the loop after sensor actuation by a threat. Without these functioning onboard cameras, however, an external camera had to be integrated into the system as a temporary solution for the prototype. This would emulate the functionality that would exist with working camera drivers.

A single-camera was utilized for the sensor field. The camera was an AXIS Model 214 PTZ IP surveillance camera with 18x optical zoom and autofocus capabilities. It had the ability to be remotely panned, tilted, and zoomed. The camera had Ethernet interface but no 802.11 capability. The camera ran an internal HTTP server on top of a Linux kernel. This HTTP server provided a streaming feed from the camera in motion joint photographic experts group (MJPEG) format, allowed for camera configuration, and for the camera to be remotely controlled through HTTP requests [56].

To bring the camera up on MSAT's wireless tactical network, the camera was connected via an Ethernet cable to a battery powered wireless access point (AP). This

AP was configured as a wireless client on the network and was assigned a fixed IP address. The camera's HTTP server was exposed to MSAT's tactical 802.11 network through a port-forwarding rule on its Ethernet-connected AP acting as a wireless client.

MSAT's relay used a loop to generate an HTTP request to the camera server, every second, in order to obtain a JPEG camera-image of the current view of the camera. This HTTP request was generated using the *Request* module. Upon receipt of the response from the camera server, the relay process converted the jpeg image to a base64 encoding, which was simply a string representation of an image, utilizing a method native to Node.js. This base64 image was then embedded into the payload of a POST request that was then sent to a dedicated HTTP server on the MSAT application server, running on a port separate from the port used to service connected clients. In this way, image streaming could be accomplished to the application server from behind the NAT of the Verizon MiFi device.

Moving and zooming the camera was accomplished through the Pan-Tilt-Zoom API on the Axis camera, which consisted of using HTTP requests to the camera server. Because the HTTP server on the camera was inaccessible behind the local NAT, users of the application outside of the local network wanting to control the camera had to traverse the NAT. This was accomplished with the following method: the client would generate a command utilizing the UI on their mobile device and this command would be sent to the MSAT application server through the WebSocket, which maintained a persistent connection between client and server. The Application server would then store this command, and attach it to the response message to the next POST request containing the base64 image from the relay running behind the NAT. Recall that this POST request containing the camera images arrived every second according to the interval set on the relay. The relay would receive this command in the response, parse it into the correct HTTP request format for the Axis camera, and send it. The camera would receive this command, and move the camera accordingly.

C. UI

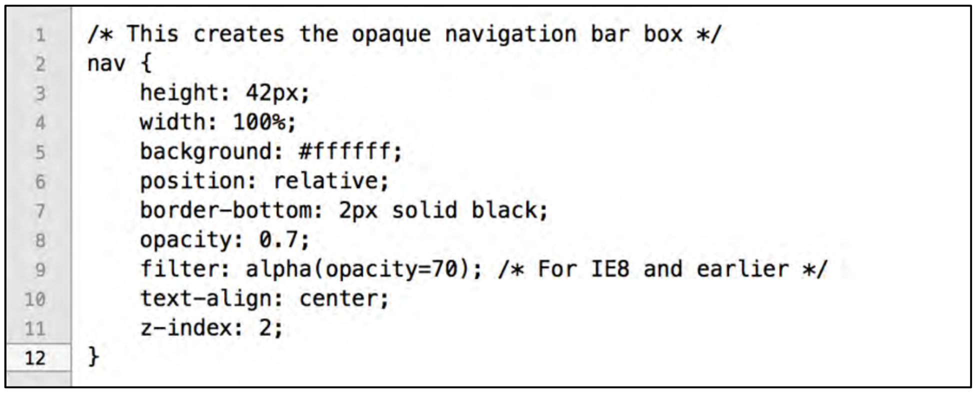
MSAT presents a web application to users' web browser windows. Here the user can interact with the different functions of the application, depending on what features the browser supports (i.e., Safari, Windows 7/8, Firefox, Chrome). Most modern web browsers try to support all the latest additions of JavaScript, CSS3, HTML5 and all the various media plug-ins; however, they all complete that task in different ways. This is an important aspect to consider when designing web applications so that a user is not penalized due to their choice of web browser. Essentially, the use of well thought-out code allows for compatibility with the largest number of browsers and devices. Such ubiquitous compatibility presents a formidable challenge, but since 2010 there has been a significant drive for developers to achieve this goal, that is, to best respond to the user's device and browser with appropriately formatted application information.

The web browser is a unique medium upon which a web developer works, since it has no boundaries like an art canvas. Instead, the x and y coordinate planes on the scroll bar are indefinite. This makes working within the medium very fluid, due to the fact that the browser can scroll and size freely. In the past, this medium for web development has been approached as if it were a print medium, with specific boundaries and limits. The print medium is not fluid and a design width and height is the first thing that is settled upon when beginning a project interface [57]. Therefore, a different approach needs to be made so that we can tackle this issue that is concisely described in [41]: "In short, we're faced with a greater number of devices, input modes, and browsers than ever before."

1. RWD Description

The basics of RWD were briefly discussed in Chapter II, but a more in depth description is needed to fully understand how it was utilized in the development of the MSAT UI. RWD is implemented through three key tools and is described in [41]. "Fluid grids, flexible images, and media queries are the three technical ingredients for responsive web design, but it also requires a different way of thinking. Rather than quarantining our content into disparate, device-specific experiences, we can use media queries to progressively enhance our work within different viewing contexts" [41]. In terms of the MSAT, the web application would already have a responsive base layer—the map with node positions, detections, and tracks. This layer could already be easily scaled, zoomed, and resized. The challenge in making this design responsive arose from the implementation of a navigation bar and the various content boxes (i.e., chat application, video feed, node information) that would be displayed.

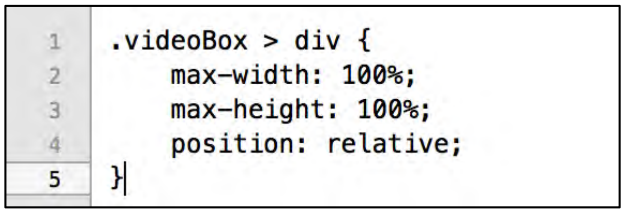
For the purposes of the MSAT application, fluid grids were not necessary since the design relied upon a navigation bar on top of an interactive map. The sole focus would be to have one element, a fluid navigation bar, able to scale to a variety of devices and provide the user with commonly utilized functions to interact with the sensor field. Figure 23 is an example of how this navigation bar's container would be styled, on top of the map layer.



```css
/* This creates the opaque navigation bar box */
nav {
    height: 42px;
    width: 100%;
    background: #ffffff;
    position: relative;
    border-bottom: 2px solid black;
    opacity: 0.7;
    filter: alpha(opacity=70); /* For IE8 and earlier */
    text-align: center;
    z-index: 2;
}
```

Figure 23. Fluid Element Example in the MSAT CSS3 File

Flexible images are achievable through fairly straightforward code, much like the concept of the fluid element shown above. The key concept is to utilize a page structure that defines containers for elements and then the elements can fill the width of their container. If the container shrinks for different screens, so does the image, hence its ability to adapt to the size. Figure 24 is an example of how the video image is defined within the CSS3 file in order to make it a flexible image.

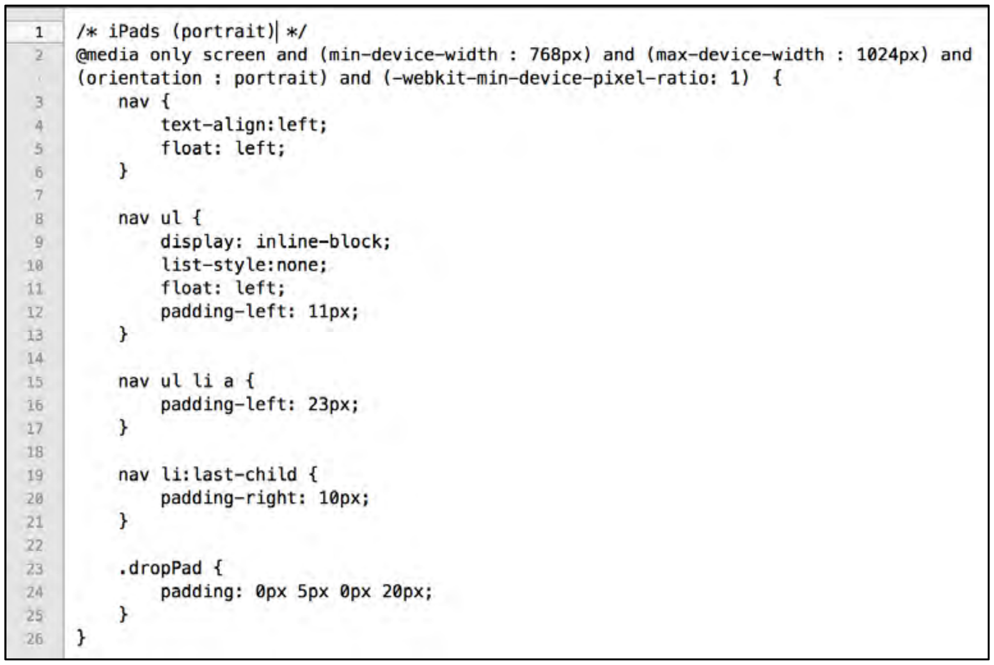


```
.videoBox > div {
    max-width: 100%;
    max-height: 100%;
    position: relative;
}
```

Figure 24. Flexible Image Example in the MSAT CSS3 File

These three lines of code allow the image to accomplish the previously defined goal of being a flexible image by creating a container (i.e., div) within the video box (i.e., videoBox) and allowing it to fill that space. If the video image size is larger than that of the video box, then the image will be cropped.

The media query contains two pieces of information: the first being the media type (i.e., screen for this case) and the second being the query of a media feature (i.e., width or height and some associated value). When the CSS3 code is executed by the browser and a media query is matched then the enclosed styling of that query will be applied. Figure 25 is an excerpt of media query CSS3 code that is used in the MSAT, which allows us to query for a mobile device screen and apply a smaller navigation bar.



```
/* iPads (portrait) */
@media only screen and (min-device-width : 768px) and (max-device-width : 1024px) and
(orientation : portrait) and (-webkit-min-device-pixel-ratio: 1)  {
    nav {
        text-align:left;
        float: left;
    }

    nav ul {
        display: inline-block;
        list-style:none;
        float: left;
        padding-left: 11px;
    }

    nav ul li a {
        padding-left: 23px;
    }

    nav li:last-child {
        padding-right: 10px;
    }

    .dropPad {
        padding: 0px 5px 0px 20px;
    }
}
```

Figure 25. Media Query Example in the MSAT CSS3 File

This media query specifically targets iPad and iPad mini, in portrait mode, by defining a specific device width and pixel density. The follow-on styling that occurs inside the media query brackets is used to fix break points that occur within that device's screen. Unfortunately, the task is not so simple as to write one media query and be done. The design has to be thoroughly tested to identify break points (i.e., content begins to degrade, overflow containers, or loose functionality) and at each point a new rule needs to be applied to fix the degradation. This is a tedious process that is time consuming for a developer. The end result is that for the extra time invested by the developer, in thoroughly testing the design, the output will be a more universally formatted tool that all devices can use.

With all these tools in concert, the application began to reflect a RWD aimed to deliver content to a wide range of devices. Figures 26–32 are screen shots of the application on a desktop utilizing different browsers, two different Android and Apple tablet devices, and an iPhone mobile browser in the landscape and portrait orientation.

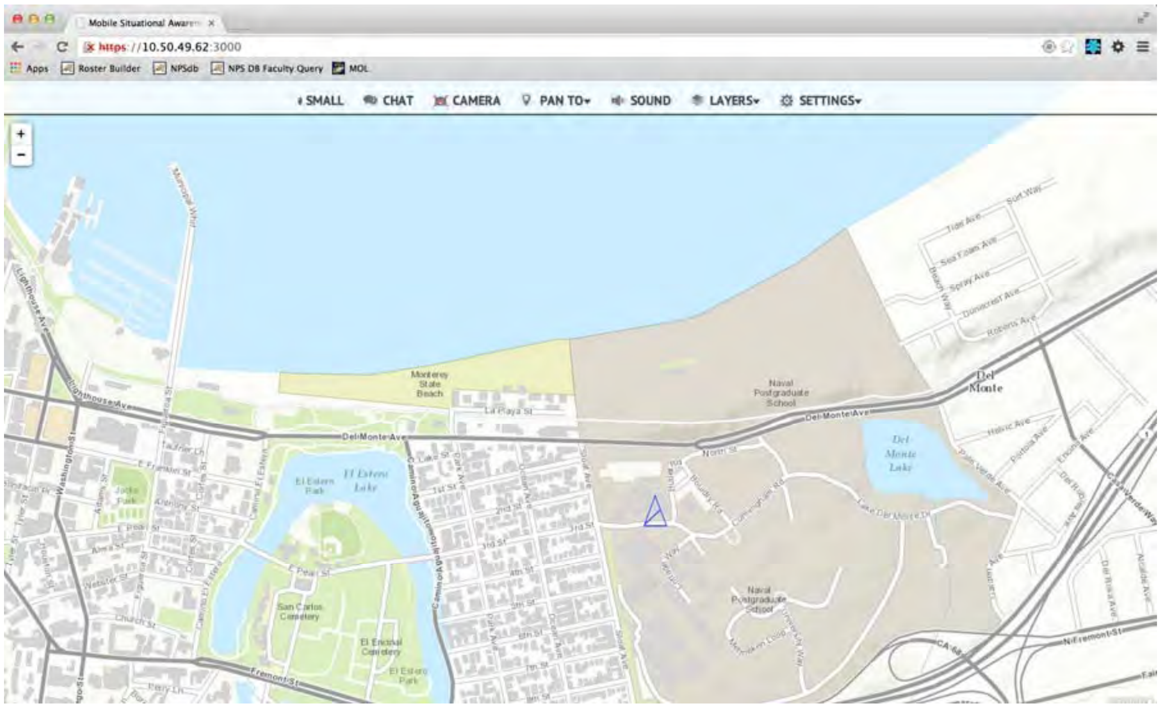


Figure 26. Desktop Device Running MSAT in Chrome Browser

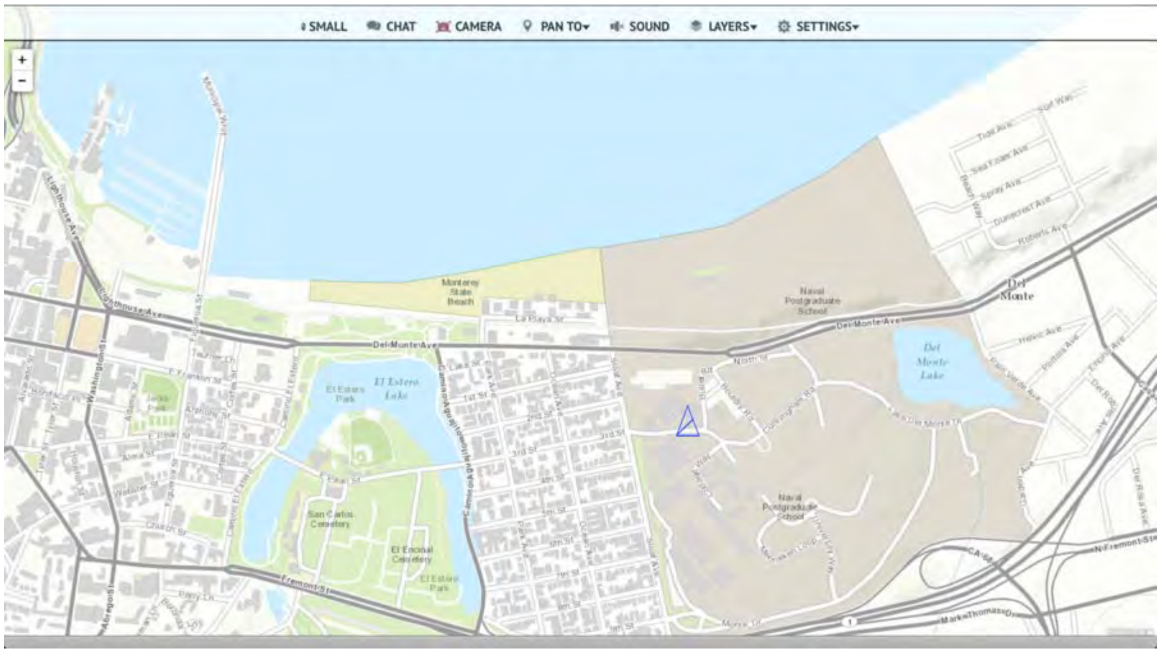


Figure 27. Desktop Device Running MSAT in Safari Browser

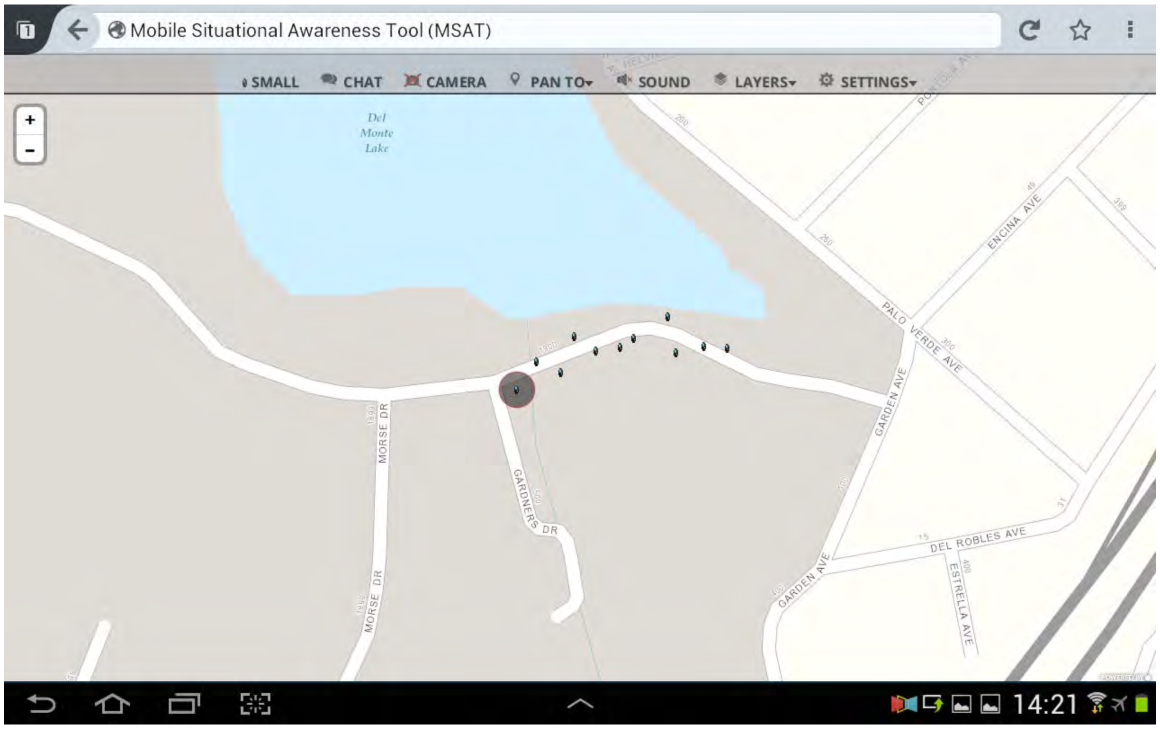


Figure 28. Android Tablet Landscape View

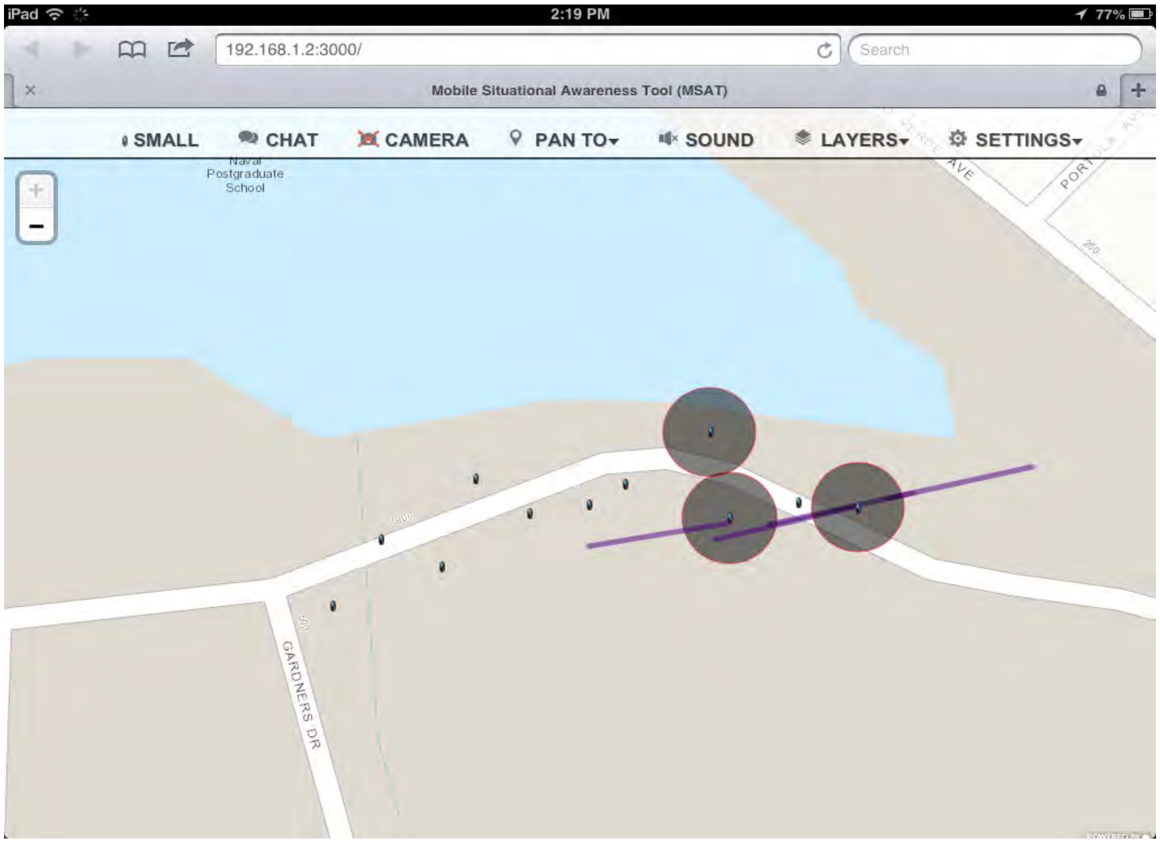


Figure 29. Apple iPad Mini Landscape View

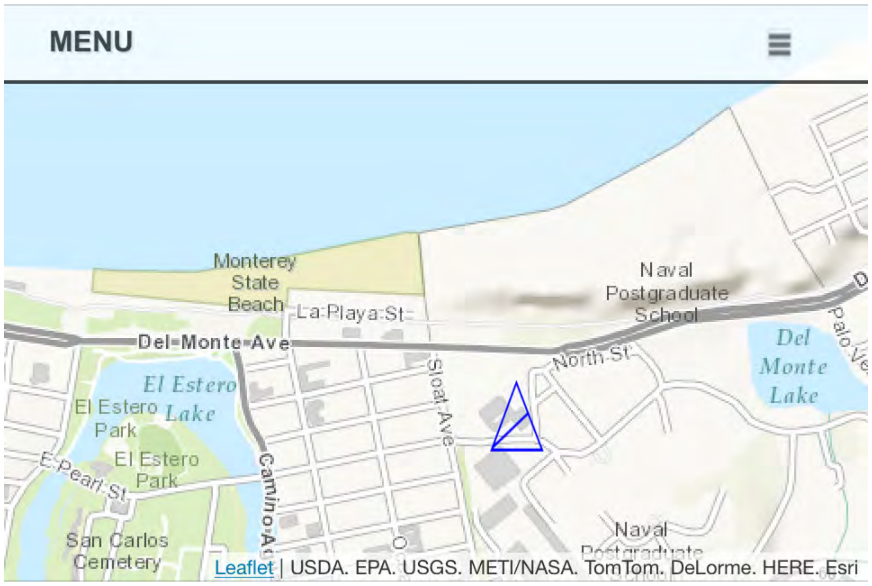


Figure 30. iPhone4 Landscape View

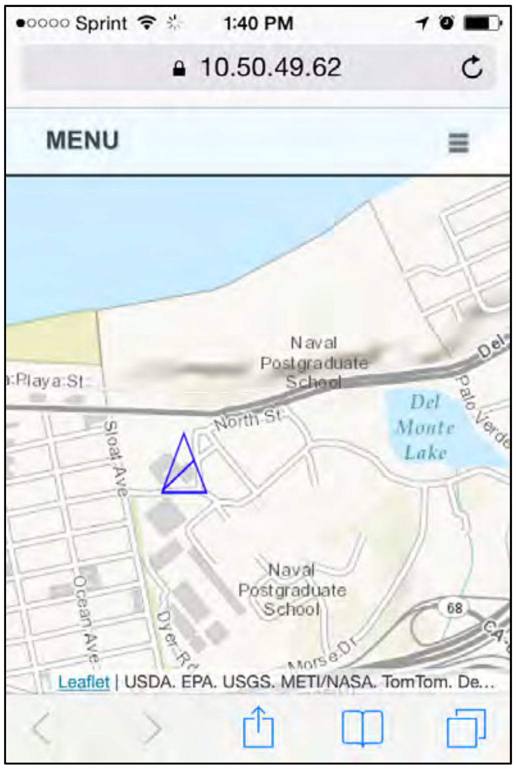


Figure 31. iPhone4 Portrait View

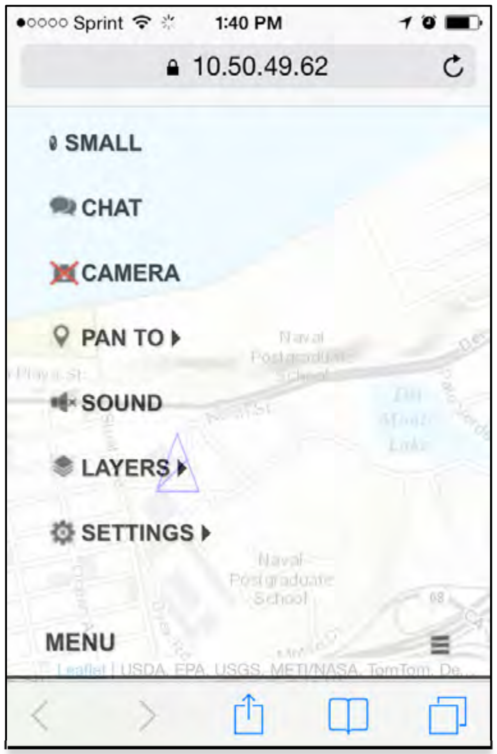


Figure 32. iPhone4 Portrait View with Menu Expanded

These screen shots of various devices and their browser highlights the RWD implementation and the motivation for pursuing this framework. With this capability, the MSAT is not pigeonholed to a specific device, operating system, or browser and it provides more flexibility to the user to determine on which platform to employ it.

2. Supporting Technology

HTML5 and CSS3 alone cannot provide all the functionality required for the MSAT UI. In order to increase user feedback through audio, visual, and haptic cues there must be outside libraries and frameworks incorporated. For the MSAT system, the following libraries were used: JavaScript, jQuery, Leaflet, and Esri Leaflet.

In the book [58], the author gives a snapshot of JavaScript's history with web applications by stating:

> The History of the web is punctuated with technological improvements. One of the earliest additions to HTML was the img element, which fundamentally altered the web. Then, the introduction of JavaScript allowed the web to become a more dynamic environment. Later, the proliferation of AJAX made the web a viable option for full-fledged applications. [58]

JavaScript has evolved into a web scripting language that provides both client-side scripts and server-side network programming. We utilized this programming language exactly for that reason, as it enables a development team to code the client- and server-side code in the same manner. On the client side, JavaScript is utilized for functions that allow the user to interact with the DOM and toggle on/off certain features on the navigation bar. It is also used to parse the JSON in order to add or update node positions, detections, and tracks onto the map layer. JavaScript also makes calls to the AJAX engine, which allows for the nodes, detections, and tracks to be dynamically populated on the map. The server-side use of JavaScript will be covered in a later section, titled *Web Application Server.*

JQuery is a library that abstracts away the details of JavaScript and purports to allow simpler, cleaner scripting on the client-side [59]. We utilized this in only a few instances, as it afforded an easier way to manipulate an item on the page versus

JavaScript. This use of jQuery can most notably be seen in the drop down menu for mobile devices (i.e., Figure 32). The function to create this effect is built from the JQuery library.

Leaflet is an open-source mapping tool that was discovered during the researching phase of the thesis. The website [60] states,

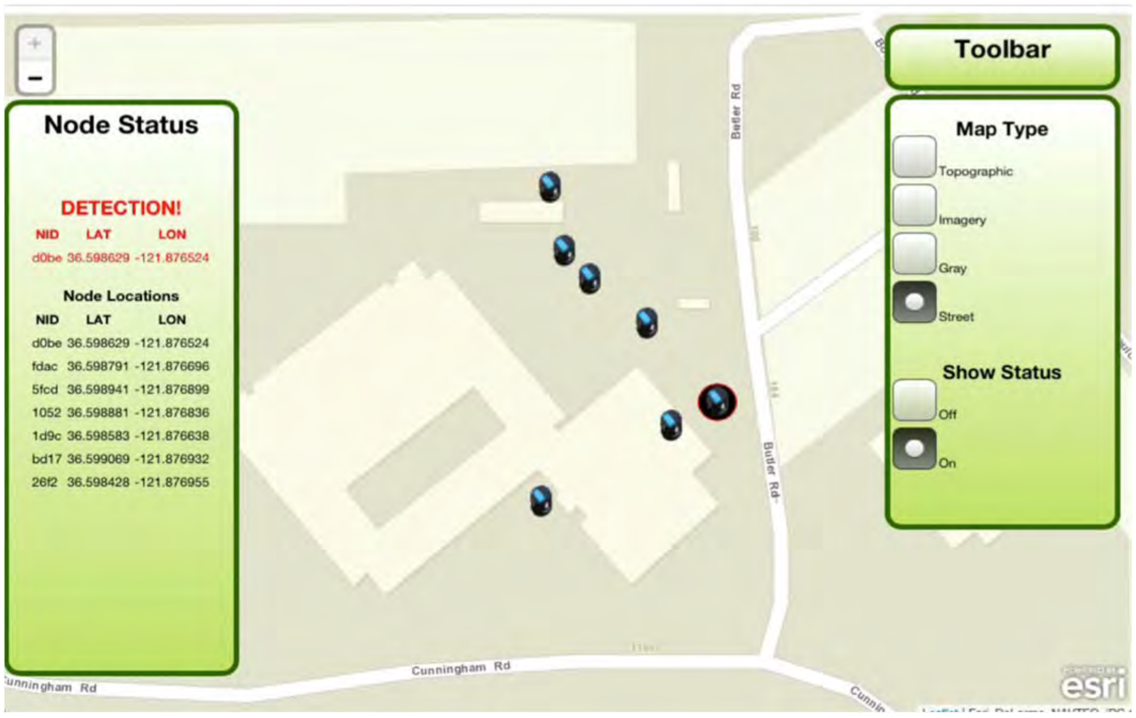
> Leaflet is a modern open-source JavaScript library for mobile-friendly interactive maps…Leaflet is designed with *simplicity*, *performance* and *usability* in mind. It works efficiently across all major desktop and mobile platforms out of the box, taking advantage of HTML5 and CSS3 on modern browsers while still being accessible on older ones. [60]

This was a suitable fit with the rest of the web application, due to its availability, development community, and ease of use. It allowed for a rapid prototype of MSAT to be developed in the early stages of the thesis process. Leaflet allows for the loading and placement of map tiles within the browser and the control over styling and size for the node icon, detection radius, and track segment.

Esri Leaflet is an open source plug-in that allows for Esri's ArcGIS map tiles to be easily displayed and manipulated within the Leaflet library [61]. This provided the MSAT application with various high quality maps from which the operator can select, while keeping with the requirement of finding open source solutions to minimize cost.

3. Testing

Designing and developing the UI was approached in two phases. The objective of the first phase was to create a simple, yet functional, UI prototype that would allow for the testing and development of the web application server. Since the majority of the intensive coding work revolved around the server, it was important to have a rapidly developed (i.e., low development cost) client-side device that could portray an adequate picture of the information being retrieved and parsed by the server. Figure 33 shows a screen shot of the UI prototype; this UI was utilized for the majority of the development of the web application server.


Node Status

DETECTION!

NID	LAT	LON
d0be	36.598629	-121.876524

Node Locations

NID	LAT	LON
d0be	36.598629	-121.876524
fdac	36.598791	-121.876696
5fcd	36.598941	-121.876899
1052	36.598881	-121.876836
1d9c	36.598583	-121.876638
bd17	36.599069	-121.876932
26f2	36.598428	-121.876955

Figure 33. MSAT UI Version 1

The initial MSAT design resulted in a wide variety of ways in which the users access the application depending on the type of device used. Due to this variety, the user struggled to receive an accurate picture of the sensor field if they were not on a system similar to the development platform. In addition, interaction with this design was difficult to navigate on a mobile device. This further solidified our drive to pursue a responsive web design that could provide a consistent interface for a large variety of the current devices and web browsers.

The objective of the second phase was to build upon lessons learned during the first phase and implement a design that was unobtrusive to the user's interaction with the device. Specifically, we sought to maximize their situational awareness, make reoccurring tasks readily accessible, and provide various forms of feedback to aid in their task of monitoring the sensor field. Figure 34 shows a screen shot of the UI developed according to a RWD approach utilized for the final stages of development and for all of the field-testing.

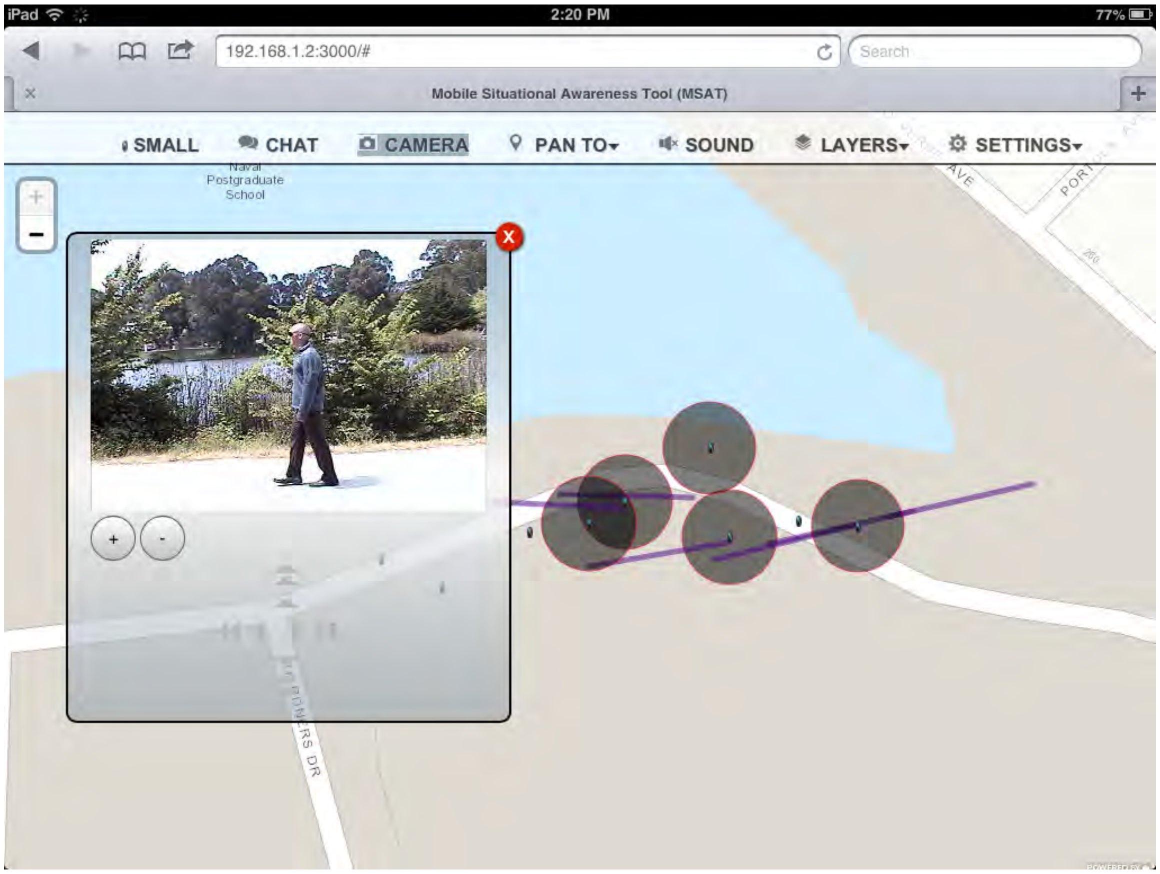


Figure 34. MSAT UI Version 2 with a RWD Approach

The UI developed according to the RWD approach was able to display the web application on a variety of devices in a manner that retained all of the application capabilities and functionality. Testing this RWD-based implementation on a multitude of mobile devices, browsers, laptops, and operating systems verified the design was operating as intended. Feedback from user interaction with the MSAT design is covered in the section titled *Discussion of Results*.

D. COMPLETE SYSTEM TESTING

1. Summary of Testing

For the final stage of the incremental testing, we spent two days testing a deployment of the complete MSAT system. The purpose of this testing was to demonstrate that such a system could feasibly automate the LP/OP, thus enabling follow on research and eventual development of a refined system for use by military end-users.

To test the system, we created a mock defensive scenario that highlighted a chosen use case, and we tested the performance of MSAT in handling the requirements of the scenario. The system was deployed outdoors in an environment that was only partially controlled, meaning the system had to operate in contention with unpredictable factors. Marine infantry officers, with combat experience in Afghanistan and no prior experience or training with MSAT, operated the system. The results of the test indicate that while the system needs further refinement and development, overall the reference design succeeded in meeting the requirements of an automated LP/OP.

2. Concept of Testing

To test the system, we first created a scenario that would highlight a likely use case for a unit employing the system in combat. Since LP/OPs are very commonly deployed in defensive operations, we decided to employ MSAT as part of a defensive operation. Specifically, MSAT would be integrated into a hypothetical defensive scheme of maneuver for a platoon tasked with blocking from a battle position. The system would be used to surveil a section of road not visible from the defensive position due to intervening terrain (i.e., a hill). An operator, without the ability to directly see the ground being covered by the sensors, would monitor MSAT for enemy activity. The effectiveness of the system would be judged by its ability to detect enemy intrusions into the assigned Tactical Area of Responsibility (TAOR), enable the operator to classify the threat with enough detail to generate a SPOTREP, and allow the operator to send the SPOTREP via the built-in form and chat application in a timely manner.

3. Conduct of the Test

We decided to utilize five Marine officers as the operators of MSAT for system testing. All had at least four years of military service, combat experience in Afghanistan, and little to no experience with employing sensor systems in combat. We sought out these personnel because of their familiarity with LP/OPs, their ability to provide feedback on the utility of such a system, and their lack of specialized sensor training. In no way were the operators evaluated, as the focus of the test was strictly the ability of the system to support LP/OP operations.

a. Scenario

We formulated a mock scenario to frame the system testing. The scenario involved establishing an LP/OP as part of a platoon in a static defensive position. The LP/OP would be responsible for reporting all movement (enemy, friendly, and neutral) within its assigned TAOR.

The following fragmentation order was created to describe the scenario and was briefed to our operators before testing began:

> **Situation:** You are a fire team leader in 3rd Sqd, 3rd Plt, Company B, 1/3. You are currently conducting defensive operations with your Plt, in an effort to secure the town of Del Montia, while humanitarian assistance is provided to the inhabitants of the town, in order to prevent enemy insurgents from entering the city and disrupting the humanitarian efforts.
>
> **Mission:** Screen to the north of the Plt's position in order to provide early warning to the Plt of enemy approach along ASR Red.
>
> **Execution:** Utilize MSAT as an automated LP/OP in support of your platoon's defensive scheme of maneuver.
>
> **Administration/Logistics:** You will utilize a ruggedized tablet, in conjunction with the tactical network provided by MSAT.
>
> **Command/Signal:** Upon enemy contact, generate and send a SPOTREP to the platoon CP utilizing MSAT's integrated chat application.

b. System Setup

The setup of the testing site is depicted in Figure 35. The TAOR for the LP/OP was an east-west running road and trail bounded by a lake to the north, located on the campus of the Naval Postgraduate School. Foot and vehicle traffic in this area would be canalized due to the restricted nature of the terrain—the TAOR was bounded to the north by a lake and to the south by wooded terrain. Therefore, the most likely avenue of approach for any enemy traveling through the AOR would be the east-west running road and trail. The hypothetical platoon battle position was located one hundred meters south of the TAOR for the sensors, behind a hill with a clump of trees, such that the TAOR would not be visible from the main battle position.

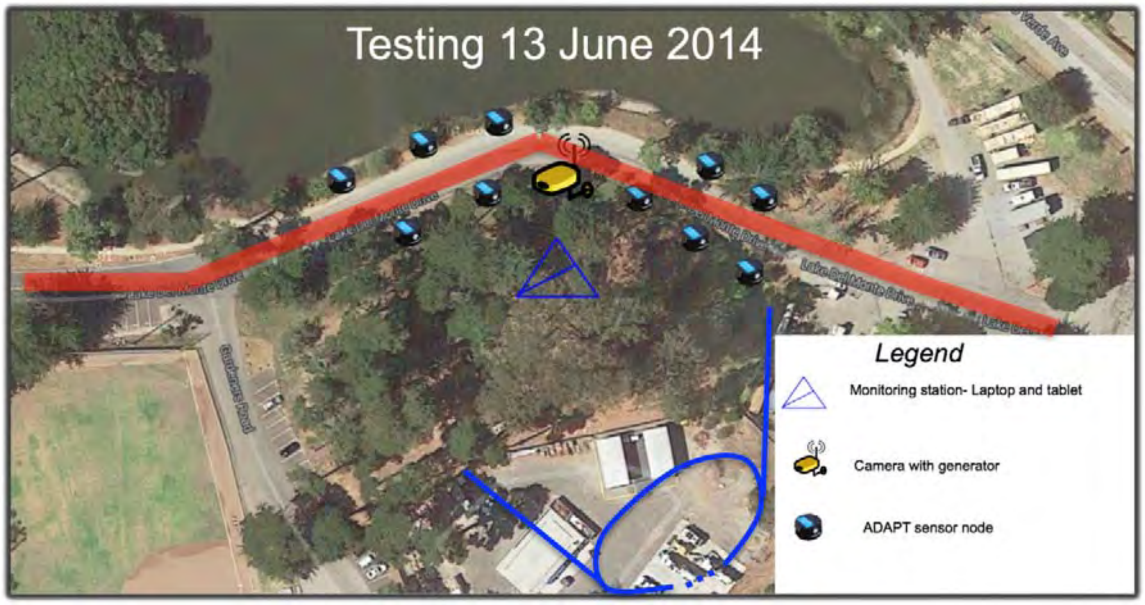


Figure 35. Testing Site

(1) Node Placement and Spacing

To enable monitoring of this most likely avenue of approach in the TAOR, 14 sensor nodes were emplaced along the road. Nodes were positioned on both sides of the road and trail for a total frontage covering approximately 200 meters. All nodes were emplaced by a single individual who carried the nodes in a specialized load-bearing pack designed for carrying antelope by hunters, shown in Figures 36 and 37. The nodes were spaced an average of 25 meters apart laterally along the road, and spaced approximately 20 meters apart (the width of the road) across from each other on either side of the road.

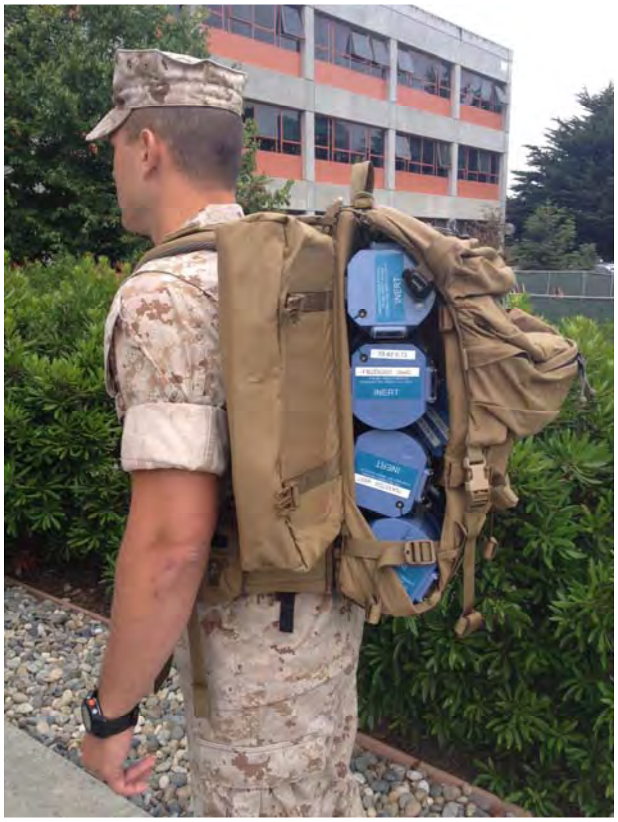


Figure 36. Side View of Nodes in Load-Bearing Pack

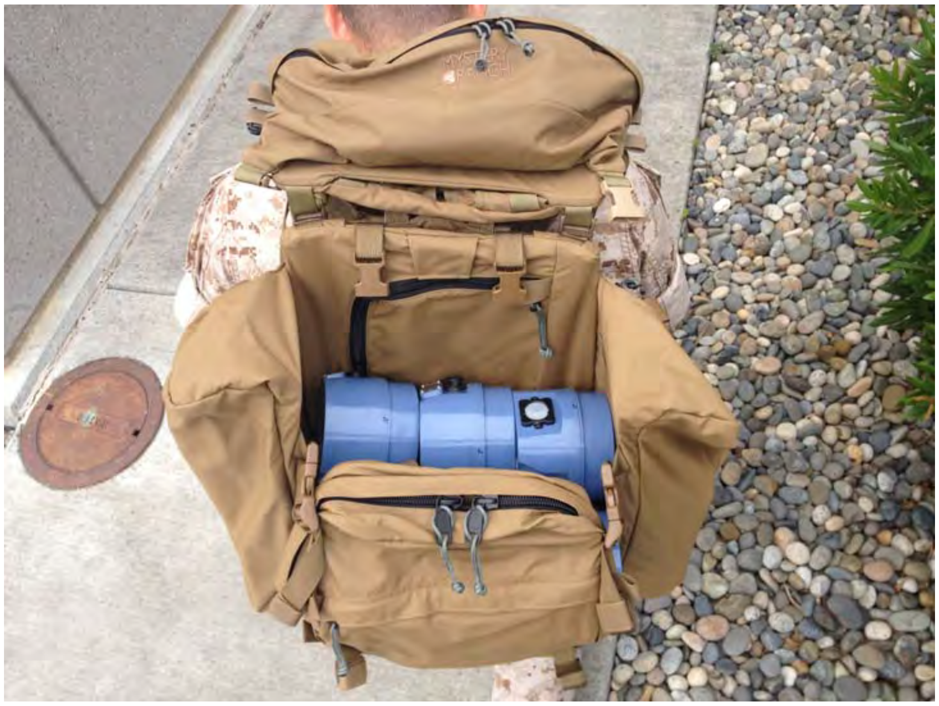


Figure 37. Top View of Nodes in Load-Bearing Pack

(2) Use of Sink Nodes

Two of the 14 nodes emplaced were configured as sink nodes. The role of these sink nodes, in addition to acting as sensors themselves, was to relay data from the non-sink nodes that were communicating via the SAS ground-radio protocol, to the monitoring station attached to the 802.11 wireless network, essentially acting as a bridge between the SAS and Wi-Fi networks. The purpose of using two sink nodes instead of connecting all of the nodes to the 802.11 was to limit traffic intensity on the wireless network that would affect application performance, conserve power on the majority of the nodes by turning off their Wi-Fi radios, and to circumvent the limitations of the 4G/LTE Hotspot. The Hotspot was limited to only 10 simultaneous Wi-Fi connections and had only a limited range (approximately thirty meters) that could not fully encompass all of the nodes, especially the ones at the outer edges of the sensor field.

(3) Camera Placement

The AXIS Camera used for visual over-watch of the sensor field was the only system component that required an external power source. Thus, the camera was powered via a 100W inverter, plugged into the cigarette lighter of a pickup truck; the camera was placed on top of this truck. The truck-mounted camera was positioned in the center of the TAOR (Figure 35) such that it would be capable of observing down the eastern and western avenues of approach. The camera was connected via Ethernet cable to a portable, battery powered 802.11 access point configured to act as a relay between the camera's video server and the 4G/LTE Hotspot. Utilizing this wireless relay, the camera's server was connected to the local Wi-Fi network and assigned a dynamic IP address via the 4G/LTE Hotspot's DHCP server, making camera controls and video feed accessible from the rest of the network.

(4) Tactical Wireless Local Area Network (TWLAN)

In order to transition the MSAT from the lab to an outdoors testing area, there needed to be a mobile network in place. MSAT components were networked utilizing 802.11 g. A Verizon 4G/LTE Hotspot (MiFi 5510L) provided the Wi-Fi bubble for the sink nodes, the camera, the application server, and the mobile monitoring devices to communicate and also access to the outside Internet via Verizon's 4G infrastructure, which provided coverage at the testing area. The Wi-Fi was secured using WPA2 with a pre-shared key (PSK). All connected components, to include the ADAPT nodes, utilized this key to gain access to the network.

(5) Application Server

The application server was installed on a laptop located inside the cab of the pickup truck, located with the Hotspot and the camera. The laptop was powered with its own internal battery.

(6) Monitoring Devices

An iPad Mini tablet with a ruggedized case and a Samsung Galaxy Tablet were used for the monitoring platforms.

4. Testing Protocol

A testing protocol was created in order to test the system's capabilities. Actors would walk tracks through the sensor field, and mobile devices connected to the application server were utilized to classify intrusions, generate SPOTREPs, and communicate the SPOTREPs over the network to the simulated platoon command post connected via 802.11.

a. Simulated Intrusions into the Sensor Field

In order to test the ability of the system to detect intrusions and enable the classification of threats and the generation and sending of SPOTREPs, events were randomly generated and presented to the system. A random event generator was created in the Python programming language that when executed output an event description that would define the SPOTREP of the event. The events were limited in scope in order to facilitate standardized reporting using a prepopulated SPOTREP form within the MSAT user interface. All events would be classified according to the acronym SALUTE. For the field testing, the size of the event was limited to either one or two personnel, the activity of the personnel was either running or walking, the location of the personnel was either from the west or from the east, and the unit identification was friendly, enemy or neutral. Time was the time of observation in local time, and equipment was a rifle, a pistol, or nothing at all. Finally, no-events were possible, meaning no intrusion would occur during a time period.

Actors were utilized to play out the randomly generated events for the system to attempt to classify. Actors simulating friendly forces wore digital USMC camouflage utilities. Actors wearing green flak jackets simulated enemy forces, and neutral forces wore neither. The actors carried rubber rifles, pistols, or nothing based on the generated scenario. The actors entered into the sensor field on the path either from the west or the east depending on the scenario, and ran or walked according to the event description.

Tests were run for approximately four hours each day for two days, and this total testing time was subdivided into 5-minute periods. One event would be executed every period, with no activity being a possible event. Each operator was presented with at least three events.

b. Use of Mobile Devices to Monitor the Sensor Field

Operators were tasked with utilizing a tablet to monitor the sensor field and generate SPOTREPs based on intrusions into their TAOR. An Android tablet would be utilized for half of the events, while the other half of the time an iPad Mini would be used. Before testing began, the operators were given five minutes to familiarize themselves with the user interface and to practice controlling the camera.

Constantly running a camera on the ADAPT sensor nodes would consume battery power at a rapid rate, thus limiting its lifespan in the field before needing to be recharged. Also, constantly monitoring a camera requires persistent human attention, thus consuming the human resources of the combat unit utilizing the system. Therefore, the constraint was imposed that the camera could only be utilized by the operator to *classify* threats. In other words, the camera could only be utilized by the operator after the sensors reported an intrusion into the sensor field through the system’s reporting of detections and tracks to the monitoring device. The camera would be the primary means of classifying threats due to the lack of a mature threat classification algorithm within the sensor system because of the lack of a multiplicity of sensing modalities.

Every event during system testing was divided into three phases. During Phase One, the threat made entry into the sensor field from the east or the west and walked a track along the path. The distance into the sensor field that a threat was able to move before the operator made the determination that an intrusion had occurred based on reporting from the sensor nodes was recorded for every event. Upon detection of the threat, Phase Two began and the operator was allowed to open the camera interface on the tablet and attempt to classify the nature of the intrusion through control of the camera. The time from threat detection until threat classification, using the camera, in seconds was recorded as a metric. After the threat was classified, Phase Three began, which was the generation and sending of the SPOTREP. The amount of time taken by the operator to open the SPOTREP form, complete the form, send the form, and have it received by the mobile device simulating the platoon CP was recorded. Finally, the accuracy of the SPOTREP submitted by the observer was recorded.

5. Limited Scope of Testing

The purpose of the testing was to demonstrate the potential for a reference design for an automated LP/OP and not as a test to generate precise metrics for a finished, production product. Therefore, environmental factors were not precisely controlled and results would be skewed by human factors such as unique characteristics of the individuals monitoring the system through the UI, and by environmental factors such as wildlife in the testing area, weather conditions, vegetation, and unplanned movements of people through the site. The uncontrolled and unpredictable factors were not avoided; rather, they were embraced for the interesting commentary they would provide regarding the feasibility of the system.

6. Discussion of Results

a. Node Emplacement

The nodes were hand-emplaced along the route by a Marine carrying all 14 nodes in a pack. The 14 total nodes weighed 43 pounds, and it took 7 minutes and 38 seconds for them to be set out the first day, and 6 minutes and 44 seconds to set them out the second day.

b. Network Formation

The individual sensors were always emplaced while in the "off" state, and we observed how long it took on both days for the network to autonomously form after turning on the nodes. Across both days, it took an average of 27 seconds for the first link to be created from a sink node to the application server, but 15 minutes and 30 seconds for the entire network to form. We counted the network as being formed after the majority of nodes obtained GPS fixes and created links with neighbor nodes, and no new links were in the process of being formed. The first day, 12 of the 14 nodes joined the network, with the remaining two not forming links with neighbors. This occurred both times the network formed. The second day, only 11 nodes initially joined the network. After the network stabilized, meaning at least five minutes had passed with no new links being formed, we set out an additional sink node and three non-sink nodes. It took an additional 8 minutes for three of these four new nodes to form links to neighbors, with one never forming any links and joining the network. In total, four of the seventeen nodes put out never joined the network, which could be due to hardware failures that have been present throughout the testing process.

c. Intrusion Events and Generating SPOTREPs

Five separate Marine Corps officers operated the system during the field experiment to test MSAT's ability to respond to various types of intrusions into the TAOR of the LP/OP. We generated SALUTE events, with our Python program, to present the operators with scenarios over the five-minute test period. Upon the intruder entering into the sensor field, each operator had to rely on the sensors in order to determine when an intrusion had taken place in their TAOR. In Figure 38, the initial detection in the sensor field is shown, as it was displayed to the operator with a mobile device.

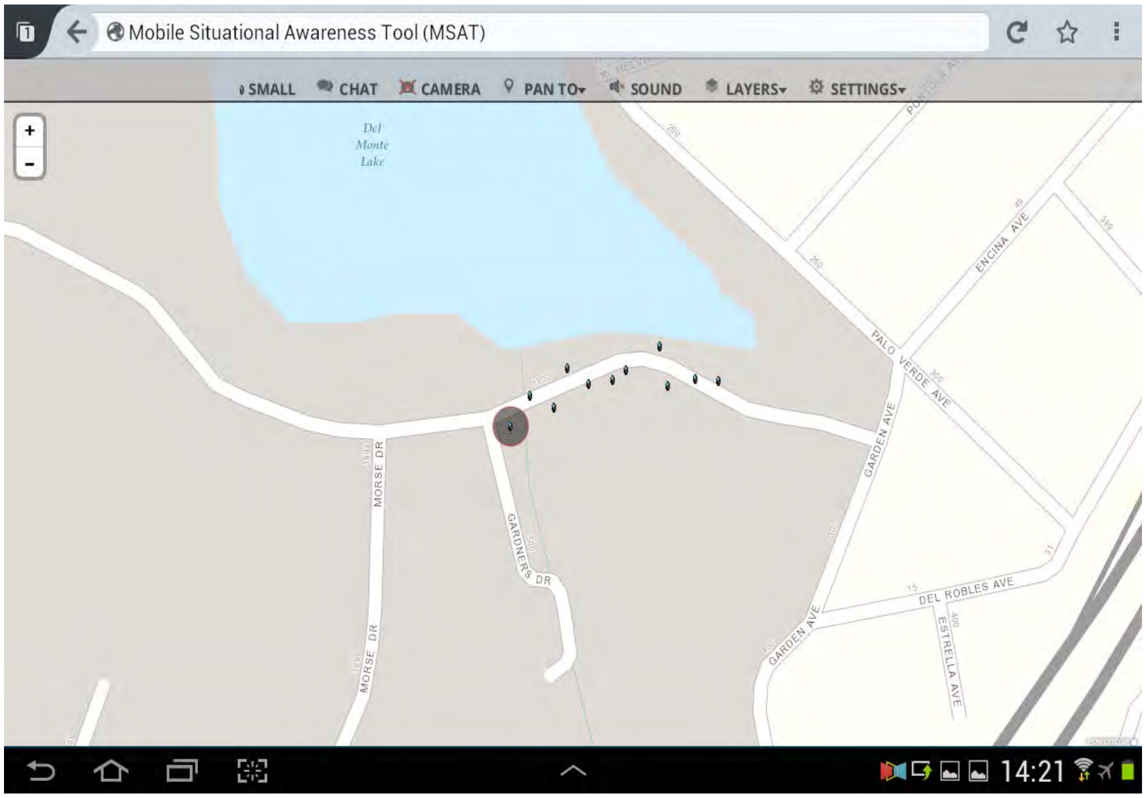


Figure 38. Android Tablet Depicting First Intrusion Detection from the West Side of the Sensor Field

Once the operator observed the initial detection, they would begin to focus more intently on the area for supporting evidence of an intruder. The follow-on progression of an intruder continuing into the sensor field, from initial detection, is displayed in Figure 39. They could utilize the detection and/or the tracks to make this determination. The majority of the operators waited until tracks formed, which, according to the tracking algorithm of the system, were the result of three simultaneous detections in close proximity. The population of a track, after three detections, is shown below in Figure 40. This strategy of relying on tracks required that the user focus less overall attention on the screen constantly and decreased the incidences of false positives for intrusion. There were two false positive intrusions identified during the two days of testing: a flock of geese in the TAOR caused one, and the other was likely due to vegetation moving in the wind that caused PIRs on the sensors to trip in the absence of human intruders.

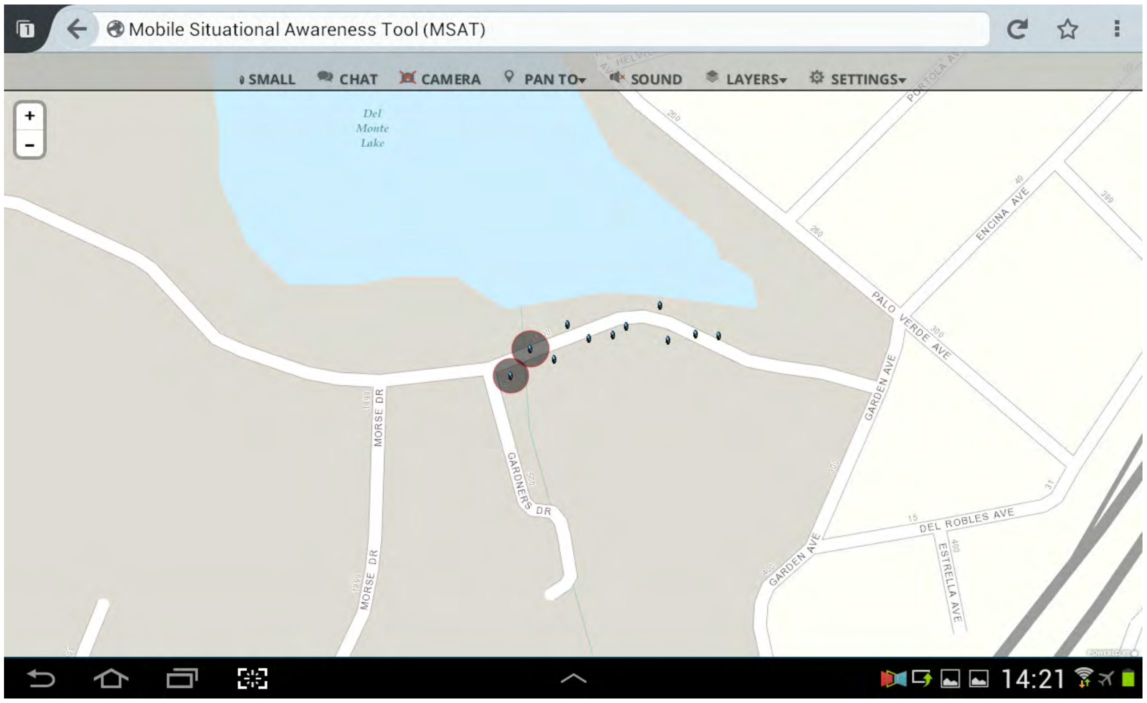


Figure 39. Intruder Continuing into the Sensor Field and Generating Second Detection

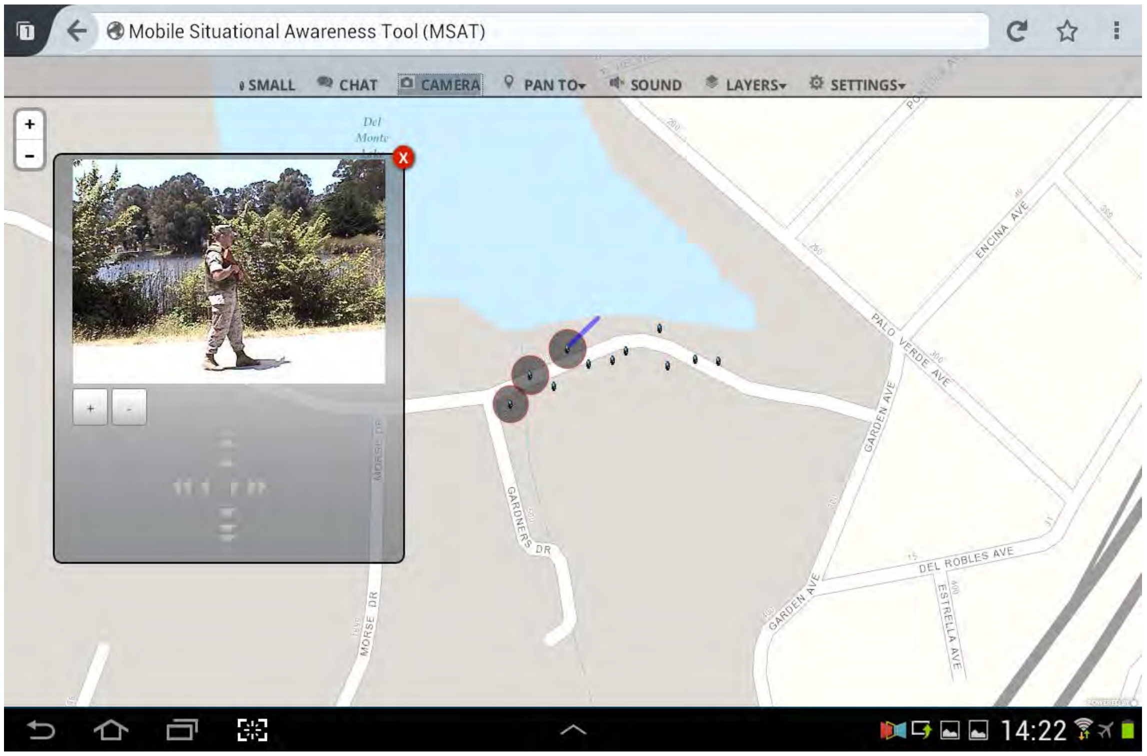
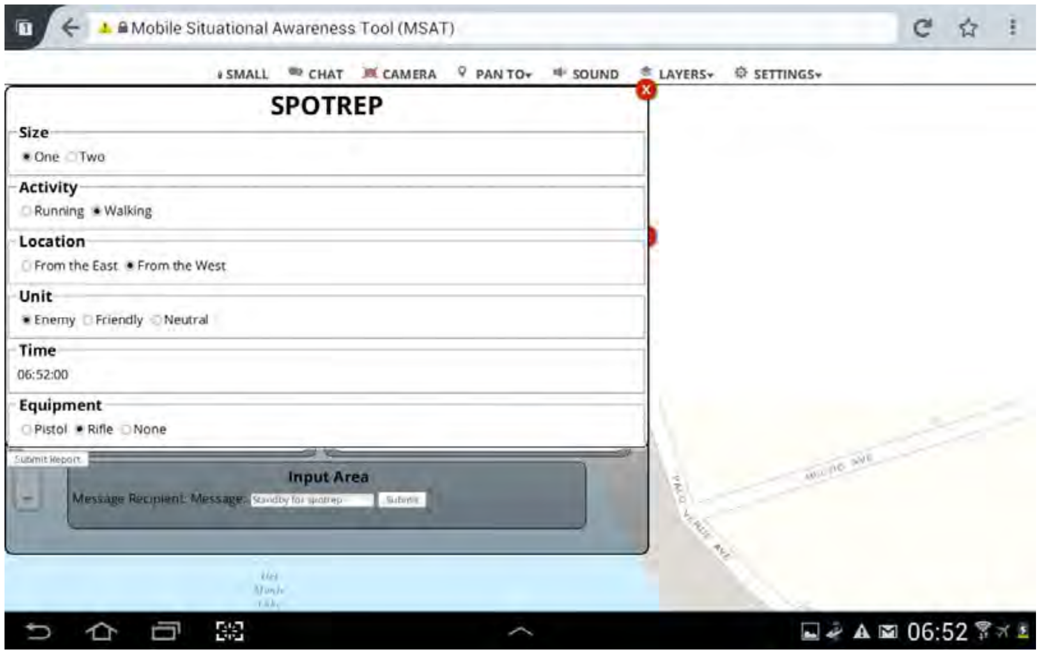


Figure 40. Intrusion Classified Using the Camera After Generating Third Detection and the Track Indicator

Over the course of the two days of testing, we recorded how far into the sensor field an intruder was able to travel before the operator identified that an intruder had entered his TAOR. Without a large variance, targets traveled on average 61 meters into the sensor field before being identified as intruders by MSAT operators. Only one event resulted in the intruder making it through the entire sensor field without being identified by the system.

For each SALUTE event, the operator opened the camera window (shown in Figure 40) to classify the threat after identifying that an intruder had entered the TAOR. The length of time it took the operators to orient the camera on the intruder and glean enough information to generate the SPOTREP was generally more varied than the time to detect the initial presence of the intruder. Sometimes the operator was able to quickly and efficiently manipulate the camera and find the target in less than 10 seconds. Other times they fumbled with the controls—one operator took 37 seconds to get the camera trained correctly on the target, and there were three instances of the operator not being able to identify the intruder after initial detection. The rest of the twenty-two events resulted in the success of the operator in classifying the target. There did not appear to be any correlation between the speed of the target and the success of classification—two of the missed intruders were running instead of walking. However, we noticed that the users had more difficulty utilizing the iPad Mini to control the camera than utilizing the Android tablet.

Finally, we observed the users as they attempted to complete the SPOTREP form in the UI and send it to the simulated platoon command post. The SPOTREP is shown below, in Figure 41. The average length of time to complete the SPOTREPs was 11 seconds, and the average time for the SPOTREP to be received by the command post on the unused tablet was 1500 milliseconds. The population of the SPOTREP, within a chat application, and its receipt by the command post is shown in Figure 42. We also observed that the accuracy of the SPOTREPs completed by the users was very high. During only one of the twenty-five scenarios, the operator completed an incorrect SPOTREP by reporting that the intruder was coming from the East, when in fact the intruder was coming from the west. After this mistake, the operator stated that he had become disoriented.



SPOTREP

Size: [✓ One] Two

Activity: Running [✓ Walking]

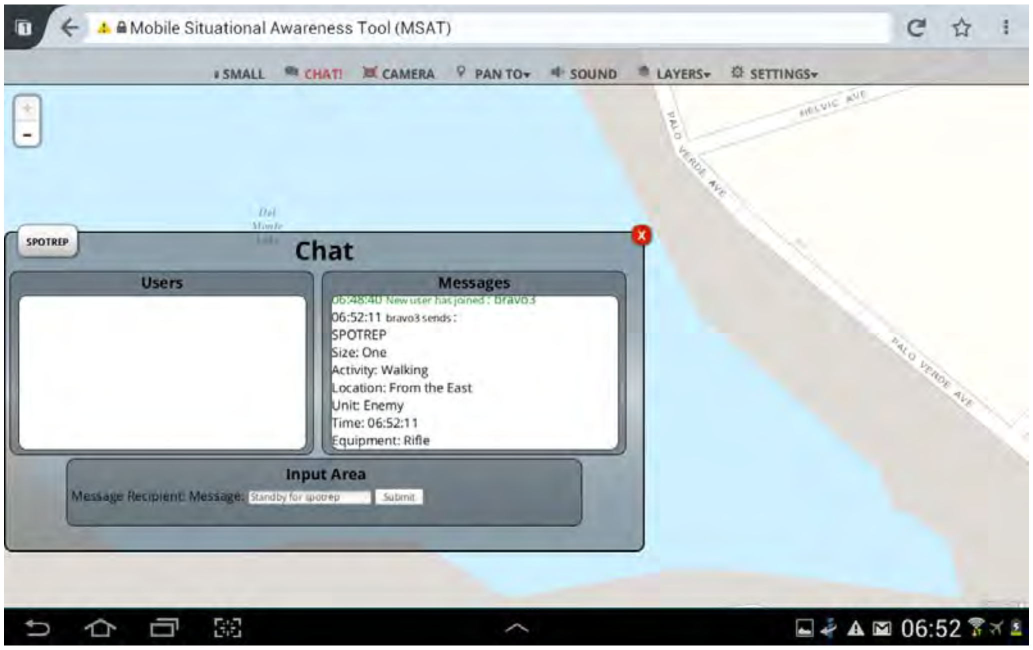
Location: From the East [✓ From the West]

Unit: [✓ Enemy] Friendly Neutral

Time: 06:52:00

Equipment: Pistol [✓ Rifle] None

Figure 41. Operator Completing SPOTREP With Form After Intrusion Classification



06:48:40 New user has joined : bravo3
06:52:11 bravo3 sends :
SPOTREP
Size: One
Activity: Walking
Location: From the East
Unit: Enemy
Time: 06:52:11
Equipment: Rifle

Figure 42. SPOTREP Sent to Higher Headquarters Via Chat Function

d. Usability Issues

We observed several usability challenges as users interacted with the UI. A major design flaw discovered was that the application screen would zoom errantly when the user attempted to manipulate the camera controls. This occurred on several occasions, and when this happened, the user would find it difficult to zoom the view back out to the original level and continue using the application. Users were also observed struggling to capture the intruder on video with the camera while attempting to classify the threat. The users often panned too far and had to pan back in the reverse direction with smaller movements under they came on target. The operators also had difficulty using the camera control buttons due to their small size and one critiqued that the opaqueness of the buttons made them difficult to see. Additionally, the radio buttons on the SPOTREP form were too small and were difficult to press by the user. Another user expressed concern that opening the SPOTREP form took two steps instead of one—first he had to open the chat widget and from there click a separate button to open the SPOTREP.

It was observed that during periods of bright sunlight, users had difficulty seeing the display on the tablet. This was an equally difficult problem for both the iPad Mini and the Android tablet. The users responded by attempting to move into the shade to reduce the glare.

e. Latency Issues

Multiple tablets connected to the application server at the same time resulted in noticeable latency. This hurt the responsiveness of the application in the hands of the operator—map tiles loaded more slowly, the camera became less responsive to user input, and the video feed would periodically go down. Interestingly, SPOTREPs being sent over the network through the chat application, which used WebSockets, did not appear to be affected.

f. Battery Performance and Power Consumption

We ran the system for four hours on both days. Every component in the system not including the camera—the nodes, the 4G/LTE access point, the application server laptop, the camera's wireless relay, and the tablets—ran this entire time using their own internal power supply. At the end of each four-hour day, the 4G/LTE Hotspot had consumed only a quarter of its battery capacity, the laptop also consumed only a quarter, and the nodes did not indicate any significant loss in capacity. The camera did not have its own battery and was run through an inverter connected to a pickup truck. The pickup truck's battery was sufficient to power the camera without the need for the truck to be started to recharge.

g. Environmental Disturbances

On the second day, a gaggle of geese wandered into the eastern portion of the sensor field and loitered on the path for about an hour. This led to almost constant detections from the nodes located in the vicinity of the geese. This initially confused the operators with a false positive intrusion, causing one operator to believe an intrusion was taking place. After using the camera to determine what this was, the operator ignored detections and tracks occurring in the area of the geese, leading him to also ignore a legitimate intruder entering the field from the area of where the geese were moving. This indicated an inability to differentiate between active wildlife and human targets using the system.

On both testing days, vehicles unrelated to testing drove on the path through the sensor field. These were not used as part of the scenarios, but we noticed that the vehicles generated noticeably longer track-lines (shown in Figure 43) than dismounted intruders due to their greater speed, allowing for easy differentiation between vehicles and dismounts.

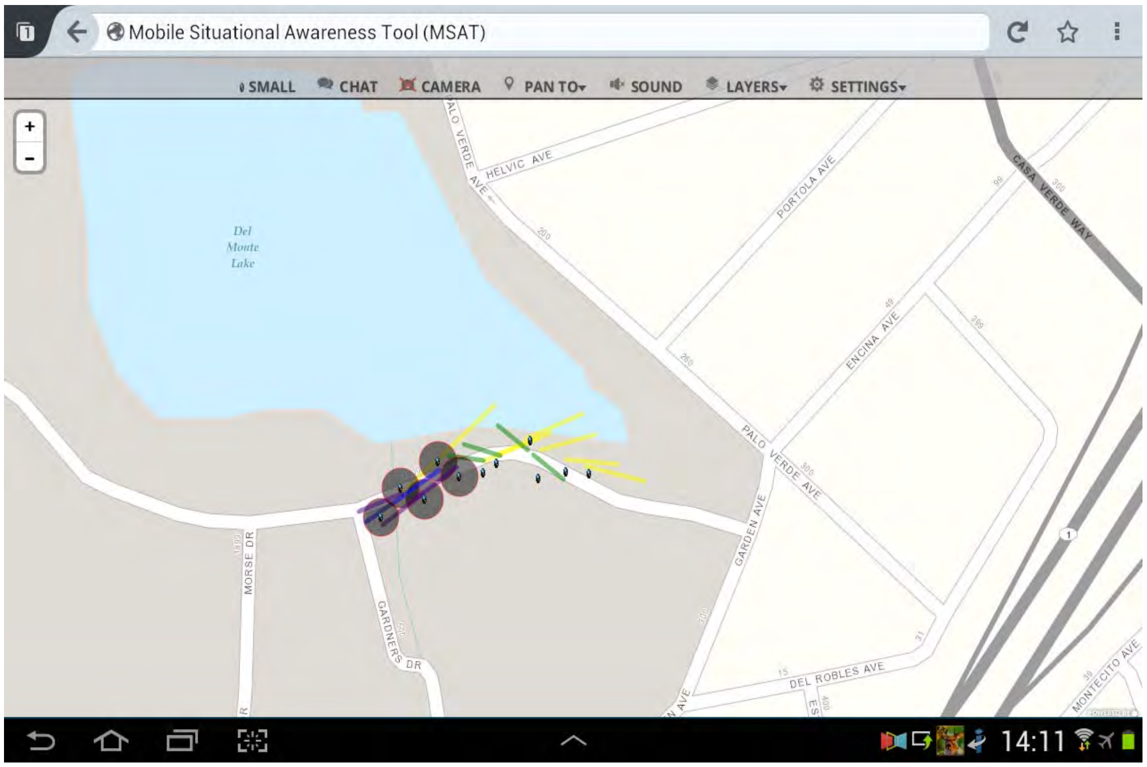


Figure 43. Multiple Confused Tracks Due to Environmental Disturbances

h. User Feedback

The system operators provided commentary on the system after use. Overall, they expressed frustration with the inability to recover from the mistake of accidentally zooming the application in too far, a problem that could be fixed by locking the display to zooming. Also, the operators identified the camera as being difficult to control with the small size of the buttons, and the cameras overall slow response to input. One operator noted that the application had the tendency to "suck" the user in, meaning that its need for careful attention caused the user to not pay attention to his physical surroundings. Additionally, an operator noted that the nodes were too heavy after being allowed to handle a sensor node. Finally, all users stated that the application would be useful to the military and expressed their satisfaction with the concept.

7. Conclusion of Field Testing

MSAT field testing highlighted key capabilities and limitations of the prototype system. The system was capable of detecting intrusions into the TAOR for the majority of events with a low false positive detection rate. However, these intrusions took on the order of several tens of meters in order to be registered by the system and alert the warfighter. Often, the system registered the intrusion after the intruder had already moved more than 60 meters into the sensor field. In a close combat scenario, this could allow enemy forces to penetrate close enough to friendly forces to throw hand grenades. An example of such a scenario would be employing sensors to protect a patrol base located in a dense urban environment.

MSAT performed during the field testing with a low rate of false positive detections, even in the face of unforeseen environmental factors such as wildlife and vehicular traffic. However, the degree of human effort required for target classification after a detected intrusion proved significant. Due to the absence of a robust threat classification algorithm, the operator had to be brought very soon into the process cycle of intrusion detection and threat classification. This means that the human had to expend much effort through the manipulation of the camera in order to determine what exactly was intruding into the sensor field. In combat, this effort would come at the expense of the situational awareness of the warfighter’s immediate surroundings. More automation in the form of a threat classification algorithm utilizing multiple sensing modalities would facilitate bringing the operator into the loop later and less frequently than with having to rely so heavily on video imagery. This would increase the combat power of the employing unit by freeing up man-hours otherwise required for system monitoring.

MSAT proved capable of autonomous network formation, but the time required to do so proved too long to be practical in combat. The network should have the capability to form in less than one minute in order to facilitate rapid deployment for the warfighter. Additionally, network latency caused unacceptable delays for the user. A TWLAN providing greater bandwidth would enable better application performance and more simultaneous users of the system.

The sensor nodes met the requirement to be mobile; a single man carried all of the nodes used for the field test in a pack. This same individual also hand emplaced the nodes. Even though a single person could carry all of the sensors, the 43-pound load should be considered too heavy for an end product. Every measure should be taken to lessen the weight burden for the modern warfighter by cutting weight in future iterations of the system. The need for a separate server laptop added to the weight and complexity of the system and adversely impacted system usability by the warfighter.

The proper placement of the nodes for maximum effectiveness was estimated based on prior testing and work with the sensor nodes, but no formal TTPs were utilized. In order to achieve maximum performance from the sensor system, TTPs should be developed that more precisely prescribe how to set up the sensor system based on the known capabilities and limitations. This would also ensure fewer gaps in coverage in the sensor field.

The MSAT performed acceptably across different platforms. Users were able to use the application on both the Android tablet and iPad, in both landscape and portrait modes, with minimal issues thus achieving the goal of cross-platform compatibility for MSAT.

Problems with the UI presented MSAT operators with difficulties that hindered overall performance. Unresponsive buttons, difficulties manipulating the map, and screen layout issues would need to be solved through several iterations of testing and development, but this is outside the scope of the current prototype.

Overall, the field testing indicated that MSAT was capable of providing a remote LP/OP functionality to the warfighter and providing a high degree of situational awareness with minimal training, even though the system had shortcomings relating to usability, power efficiency, weight, and minimal automation in classifying threats.

E. CHAPTER SUMMARY

MSAT, a reference design for an automated LP/OP, was built utilizing an existing wireless sensor node prototype and COTS components in order to provide a multi-platform system capable of providing a high degree of situational awareness for ground tactical units.

In order to build MSAT, DARPA’s ADAPT sensor node prototypes were adopted and tested by themselves. This field testing revealed that while the prototypes suffered from some severe limitations such as non-functioning cameras, they offered a functional platform for providing MSAT with its sensor capability. Not only were these nodes capable of autonomous network formation, intrusion detection, and tracking, they were designed from COTS components with the goal of being inexpensive.

MSAT’s application software was developed as the link between the sensor nodes and the warfighter and to provide additional functionality, such as chat between simultaneously connected users, blue force tracking, and reporting. The application software consisted of an application server and UI elements. The server was first implemented with Apache2; but due to limitations validated with a comparative benchmarking test, this server was substituted with the better-performing Node.js. The UI was created with HTML5, JavaScript, and CSS3. Through the implementation of RWD, the application was made to be compatible with a variety of hardware and software platforms.

Finally, MSAT’s components were connected via a Wi-Fi network enabled by a 4G/LTE Verizon MiFi hotspot and a surveillance camera was utilized as a substitute for the sensor nodes’ non-functioning cameras. The entire system was then tested in a field-environment that simulated a defensive combat scenario. The results of the testing showed MSAT capable of providing a remote LP/OP capability to a small tactical unit. However, the limitations of the system call for to future development and testing that should be completed prior to the fielding of a system meant for warfighters in combat.

V. SUMMARY AND CONCLUSIONS

A. SUMMARY

We developed a reference design for the automation of LP/OPs for use by infantrymen in combat. Specifically, the design was tailored for the use-case of supporting a rifle squad in a defensive battle position. To achieve LP/OP automation, the MSAT prototype was created through the interconnecting of multiple existing components and the creation of custom application software.

The intrusion detection functionality of MSAT was provided through the use of sensor nodes developed by DARPA's ADAPT program. These sensor nodes relied on the use of internal PIR sensors to detect objects entering the sensor field. The sensor nodes communicated wirelessly with each other and with handheld monitoring stations through low powered ground radios and a Wi-Fi network. Additionally, the nodes were programmed to accomplish autonomous network formation, which had the effect of minimizing network configurations for operators in the field. As part of normal operation, nodes shared data with immediate neighbors in a manner that allowed data to perpetuate hop-by-hop, without the need for a global addressing scheme or globally aware routing algorithms. Such data included detections (i.e., the actuation of the PIR sensors), sensor locations acquired from GPS receivers embedded in each node, and correlated object movement or tracks. Tracks represented the estimated location, direction, and distance of an intruder moving through the sensor field. The tracking algorithm, part of node capability developed under the DARPA program, was executed locally by nodes through the interpretation of multiple detections shared across neighboring nodes of the sensor field in vicinity of the detected intruder into the sensor field.

We developed MSAT's application server in software installed on a Linux laptop. The application server acted as the intermediary between the sensor nodes in the field and the operator monitoring the system. This application server received sensor data, including intrusions and tracks, via Wi-Fi from the nodes. Our application processed this data into a graphical user interface that was presented to clients through a single page web application. Connected clients could access MSAT's graphical interface using the web browsers on their devices. The web application structure of MSAT allowed compatibility with a wide range of Android and iOS devices, with minimal need for code to be tailored to any one particular platform. In addition to providing an interface from which to monitor the sensor field, MSAT tracked friendly user positions and supported chat communication between friendly units in order to facilitate coordination on the battlefield.

We developed the application server initially using Apache server. The inability of Apache server to service a large number of clients in real-time and the difficultly customizing the application for the desired use case led us to migrate to Node.js vice Apache. We conducted benchmark testing in order to compare the performances of Apache and Node.js servers and validate the decision to incorporate Node.js into MSAT. Node.js allowed for the creation of a custom application that was more precisely tailored to the requirements of the system. Additionally, its asynchronous execution model enabled improved performance.

Inoperable drivers on the sensor nodes precluded the use of the onboard cameras for classifying intrusions into the sensor field. We integrated an external surveillance camera with the ability to tilt, pan, and zoom into MSAT as a stopgap solution for the lack of an operating camera. The application server and UI were modified to be able to stream real-time video feed to MSAT operators. MSAT operators were also given controls on the UI to manipulate the camera to view threats in the sensor field.

Client devices, the application server, and sink nodes (i.e., sensor nodes that relay data to the application server) were networked using 802.11 g/n, with a Verizon MiFi access point which provided 4G/LTE for outside Internet connectivity.

To enable the remote C2 monitoring scenario, we configured the MSAT application server to relay sensor data outside of its local network to a proxy server on the public Internet. This proxy server enabled connections through SSL/TLS to clients located anywhere with Internet access.

B. MSAT PERFORMANCE

Through a series of tests and demonstrations, the MSAT reference design demonstrated the viability of an automated LP/OP tool for use by warfighters at the tactical level. The MSAT prototype showed that a system can be built that is portable, easy to use, wireless, and compatible with commercial mobile communications equipment. Most importantly, MSAT showed that an operator could successfully utilize UGS to generate an accurate SPOTREP, which is the primary performance standard for the LP/OP. The prototype also identified challenges that must be overcome in order to provide an end product that is truly usable for the warfighter, including the need for more automation in threat classification, more power-efficient algorithms, and more sensor modalities.

Stand-alone testing of the ADAPT sensor nodes showed the sensors to be usable, reasonably durable, and capable of detecting intrusions into a sensor field. The nodes also proved capable of tracking multiple intrusions at a time through the sensor field; however, the accuracy of the tracks decreased as the targets moved closer together. Additionally, threat classification was limited only to what could be gleaned from tracks. Tracks were generated solely through the use of PIR sensors on the nodes. False PIR triggers due to environmental factors (e.g., vegetation moving in the wind and heavy rainfall) degraded the accuracy of the intrusion detection and threat tracking. By including additional sensing modalities in the system, this can be reduced or eliminated completely. However, the nodes did demonstrate the capability to withstand harsh environmental conditions such as heavy rainfall and sun exposure.

The nodes had a battery life in the field of several days. Since infantry units may have to operate for months at a time in the field, in a combat environment without access to external energy sources, a lifetime in the range of days is too short and would need to be lengthened. Due to the nodes not reliably operating in a quiescent state, the lifetime could possibly be extended greatly if a low-power state was implemented.

Testing of the application server showed that the asynchronous, non-blocking model of Node.js outperformed the multi-threaded, blocking approach of Apache server.

The testing of MSAT in a field environment demonstrated that the system could allow the user to successfully generate SPOTREPs, upon intrusions into the sensor field without having to directly observe the event, thus limiting potential exposure to the enemy. The system was successfully used with both Android and iOS devices, showing cross-platform capability with a single code-base.

The stopgap solution of using a surveillance camera had the secondary effect of showing the ability of MSAT to easily incorporate new components. Additional sensors could therefore be introduced that would further enhance the capabilities, or allow the tailoring of the system to specific use-cases.

C. RECOMMENDATIONS FOR FUTURE WORK

Several improvements could be implemented to make an UGS-based, automated LP/OP more effective than the MSAT reference design as tested. Multiple sensor modalities beyond the single PIR sensor could increase tracking accuracy and increase the level of automation for threat classification. Possible sensors to be used are the geophone to measure ground movement, microphones to measure environmental audio, and magnetometers to detect metals in the area, which could classify an intrusion as a type of vehicle.

More complex threat algorithms would have to be developed to fully leverage the use of multiple sensors and make threat determinations. These robust threat algorithms would have the effect of reducing false positives and decreasing the amount of human involvement in the monitoring of the sensor field. Adding two more PIR sensors to each node would allow 360-degree coverage and prevent potential gaps in the sensor-field. Implementing a quiescent state would reduce power consumption and increase the lifetime in the field. This would require carefully controlling the duty-cycles of the sensors, such that sensors onboard each node were successively awakened as environmental noise due to an intruder increased, with only the lowest-power-consuming sensor (i.e., the PIR) remaining on in the absence of an event. Finally, creating smaller sensor nodes would reduce the combat load of employing units, and at the same time reduce the profile to more easily avoid discovery by enemy personnel.

We did not conduct testing regarding information security, or operation in an electronically jammed environment. Sensors would need to be hardened to potential attacks to include jamming for denial of service and the unauthorized access by malicious entities to the sensor node network, which would allow breaches in confidentiality and integrity of information on the network.

The need for a separate application server apart from the user's handheld device proved unwieldy and could be eliminated through the use of an HTML5/Javascript/CSS3 application that could run in the devices' headless web-engine (e.g., Webkit), which would implement a stand-alone application on the device and still allow compatibility across different device models with a single-code base. Another option would be to run the server on a number of UGS devices, allowing for redundancy and the ability for mobile devices to approach the sensor field and query it for information. Also, the need for a separate Wi-Fi access point could be eliminated through the use of Wi-Fi direct on the mobile device, or through Bluetooth communication with the nodes.

Optimal layout of the sensors on the nodes should be studied to ensure the most efficient manner for employment. Additionally, the use of unmanned aerial vehicles (UAV) for integration with the sensor nodes for added surveillance and communication capabilities should be investigated.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] U.S. Department of Defense. (2007, Dec. 3). Army soldier MIA from Vietnam War is identified. [Online]. Available: http://www.defense.gov/Releases/Release.aspx?ReleaseID=11530

[2] V. Kaul, C. Makaya, S. Das, D. Shur and S. Samtani, "On the adaptation of commercial smartphones to tactical environments," in *Military Communications Conference, 2011 - MILCOM 2011*, Baltimore, MD, Nov.7-9, 2011, pp. 2205 - 2210.

[3] *Remote Sensor Operations, MCRP 2-24B*, U.S. Marine Corps, Washington, DC, 2004, 1.1-1.6.

[4] Capabilities Development Directorate, "PowerPoint: Intel TAMCN review," July 2006, unpublished.

[5] Z. Sun et al., "BorderSense: Border patrol through advanced wireless sensor networks," *Ad Hoc Networks*, vol. 9, no. 3, pp. 468-477, May 2011.

[6] T. Williams. (2009, Sept.). U.S. Customs and Border Protection's use of technology to better secure U.S. borders. [Online]. Available: http://www.policechiefmagazine.org/magazine/index.cfm?fuseaction=display_arch&article_id=1895&issue_id=92009

[7] *Infantry Training and Readiness Manual*, NAVMC 3500.44,U.S. Marine Corps, Washington, DC, 2012, 7-20, 7-60.

[8] F.A. Yates Jr., "Diffusion and large-scale adoption of computer-supported training simulations in the military domain," M.S. thesis, Dept. MOVES, Naval Postgraduate School, Monterey, CA, 2013.

[9] M. I. Rostovtzeff, *The Social & Economic History of the Roman Empire*. New York, Biblo & Tannen, 1926, p. 228.

[10] B. Lancaster, *The American Revolution*. Boston: Houghton Mifflin Harcourt, 2001, p. 144.

[11] L. W. Grau and J. Falivene, "Mountain combat: hard to move, hard to shoot, even harder to communicate," *The Journal of Slavic Military Studies*, vol. 19, pp. 619-625, 2006.

[12] *Mechanized Infantry Squad Operations (Bradley)*, FM 7-7J, U.S. Army, Washington, DC, 1995, Section L.

[13] "An electronic picket line at the DMZ." *Newsweek*, (Apr., 1967), p. 25.

[14] J. T. Correll, “Igloo white,” *Air Force Magazine*, vol. 87, no. 11, pp. 56-61, Nov. 2004.

[15] U.S. Marine Corps and Lieutenant Colonel S.P. Callahan, *Close Air Support and the Battle for Khe Sanh*, Washington, DC: History Division, U.S. Marine Corps, 2009, pp. 9-68.

[16] P. Dickson, *The Electronic Battlefield*, Bloomington: Indiana University Press, 1976, pp. 74.

[17] P. Brush. (2006, June). Operation Niagara: Siege of Khe Sanh. *Vietnam Magazine*. [Online]. Available: http://www.historynet.com/operation-niagara-siege-of-khe-sanh.htm

[18] E. D. Haider, “Unattended ground sensors and precision engagement,” M.S. thesis, Dept. Defense Analysis, Naval Postgraduate School, Monterey, CA, 1998.

[19] U.S. Department of Homeland Security, Customs and Border Protection. (2011, Apr.). Request for information (RFI) unattended ground sensor technology. Border Enforcement Contracting Division - Mountain Branch. [Online]. Available: https://www.fbo.gov/index?s=opportunity&mode=form&tab=core&id=8613df5f5012d854474ee4030b34e510&_cview=0

[20] R. Beckhusen. (2013, Feb.). Homeland security delays plan to place sensors on U.S. - Mexico border. *Wired Magazine*. [Online]. Available: http://www.wired.com/2013/02/border-sensors/

[21] U.S. Department of Homeland Security, Office of Inspector General. (2005, Dec.). A review of remote surveillance technology along U.S. land borders. [Online]. Available: http://www.oig.dhs.gov/assets/Mgmt/OIG_06-15_Dec05.pdf

[22] F. Zhao and L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach*, San Francisco, CA: Morgan Kaufmann, 2004, pp. xiii.

[23] Strategic Technology Office. (2014, May). ADAPTable sensor system (ADAPT). [Online]. Available: http://www.darpa.mil/Our_Work/STO/Programs/ADAPTable_Sensor_System_(ADAPT).aspx

[24] M. Hewish, “Little brother is watching you—Distributed networks of miniature sensors can enhance situational awareness and fulfill many other battlefield roles,” *Janes IDR*, vol. 6, pp. 46-52, June 2001.

[25] Silicon Labs, The evolution of wireless sensor networks. (2013, May). [Online]. Available: http://www.silabs.com/Support%20Documents/TechnicalDocs/evolution-of-wireless-sensor-networks.pdf

[26] G.E. Moore. (1965, Apr.). Cramming more components onto integrated circuits. *Electronics*. [Online]. *38*, pp.1-4. Available: http://web.eng.fiu.edu/npala/eee6397ex/gordon_moore_1965_article.pdf

[27] D. Kushner. (2011, June). The making of arduino. *IEEE Spectrum*. [Online]. Available: http://spectrum.ieee.org/geek-life/hands-on/the-making-of-arduino

[28] *Arduino*. (2014, May).Getting started with arduino. [Online]. Available: http://arduino.cc/en/Guide/HomePage

[29] L. Orsini (2014, April). Easy arduino: Two projects to help you get started. *Readwrite*. [Online]. Available: http://readwrite.com/2014/04/21/easy-arduino-projects-basics-tutorials-diy-hardware#awesm=~oDxb1HpdD68pYj

[30] One of the first raspberry pi computers donated to museum. (2012, Sept.). *The Centre for Computing History*. [Online] Available: http://www.computinghistory.org.uk/news/16944/

[31] Raspberry Pi Foundation. (2014, Apr.).FAQs: what is a raspberry pi. [Online]. Available: http://www.raspberrypi.org/help/faqs/ #introWhatIs

[32] Raspberry Pi Foundation. (2014, Apr.). Web server setup and wordpress. [Online]. Available: http://www.raspberrypi.org/learning/web-server-wordpress/

[33] Raspberry Pi Foundation. (2014, Apr.).Turn your pi into a low-cost hd surveillance cam. [Online]. Available: http://www.raspberrypi.org/turn-your-pi-into-a-low-cost-hd-surveillance-cam/

[34] J.K. Ridner. (2013, Dec.). What is a beaglebone?. *BeagleBoard*. [Online]. Available: http://beagleboard.org/Products/BeagleBone

[35] Banana Pi. (2014, Apr.). Banana pi—a highend single-board computer. [Online]. Available: http://www.bananapi.org/

[36] QuattroMagic (2014, Apr.). Tech specs – Rikomagic Malaysia. [Online]. Available: http://quattromagic.com/techspecs/

[37] B. Linder. (2012, May). VIA APC: A $49 android computer with an arm 11 cpu. *Liliputing*. [Online]. Available: http://liliputing.com/2012/05/via-apc-a-49-android-computer-with-an-arm11-cpu.html

[38] Intel. (2014, Apr.). Mini pc - intel nuc. [Online]. Available: http://www.intel.com/content/www/us/en/nuc/overview.html

[39] NinjaBlocks. (2014, Apr.). Ninja blocks – about us. [Online]. Available: http://shop.ninjablocks.com/pages/about-us

[40] StrongLoop. (2014, July). Big brands rely on node for APIs. [Online]. Available: http://strongloop.com/developers/node-js-infographic/

[41] E. Marcotte. (2010, May). Responsive web design. [Online]. Available: http://alistapart.com/article/responsive-web-design/

[42] U.S. Army Signal Center of Excellence. (2011, Aug.). Army cellular capability development strategy—vision for the future of army mobile computing. SIGCoE. Fort Gordon, GA. [Online]. Available: http://www.ecrow.org/pdf/Army_Cellular_Capability_Development_Strategy_16_August_2011.pdf

[43] L. Edmond, “Mobile computing, smaller systems, bigger solutions,” *Army Communicator*, vol. 37, no. 2, pp. 12-13, 24, 32, June 2012.

[44] *Fire Support Coordination in the Ground Combat* Element, MCWP 3-16.6, U.S. Marine Corps, Washington, DC, 2001, p. 1-1.

[45] *Combat* Stress, MCRP 6-11C, U.S. Marine Corps, Washington, DC, 2000, pp. 61.

[46] J. J. Harris and D. R. Segal, “Observations from the Sinai: the boredom factor,” *Armed Forces & Society*, vol. 11, no .2, pp. 235-248, Dec., 1985.

[47] *Scouting and Patrolling*, MCWP 3-11.3, U.S. Marine Corps, Washington, DC, 2000, 6.1-6.2, 14.1.

[48] *Marine Rifle Squad*, MCWP 3-11.2, U.S. Marine Corps, Washington, DC, 2002, pp. 5209.

[49] F. Dupont and C. Dean. (2003, Dec.). Hydration and the modern warrior’s load. Presented at RTO Human Factors and Medicine Panel (HFM). [Online]. Available: http://natorto.cbw.pl/uploads/2004/7/MP-HFM-086-$$ALL.pdf

[50] J. Wolf. (2011, July). Responsive HTML5 apps: Write once, run anywhere? Where is anywhere? [Online]. Available: http://www.wired.com/2013/11/responsive-html5-apps-write-once-run-anywhere-where-is-anywhere/

[51] N. Bevana, J. Kirakowskib and J. Maissela, “What is usability?” in *Proceedings of the 4th International Conference on HCI,* Stuttgart, Germany, 1991.

[52] J. J. Garrett. (2005, Feb.). AJAX: A new approach to web applications. *Adaptive Path*. [Online]. Available: http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications/

[53] T. Hammel and M. Rich, "ADAPT smart munitions: Summer camp final demonstration," presented at Naval Postgraduate School, Monterey, CA, Sept. 26, 2013, PowerPoint pp. 3,7,9,16-17, 19-22, 25-26, 28-29,33.

[54] Developer Network. (2014, June). Snapdragon S4 plus MSM8960 MDP/S mobile development platform/smartphone. *Qualcomm*. [Online]. Available: https://developer.qualcomm.com/mobile-development/development-devices/snapdragon-s4-msm8960-mdps

[55] G. Harutyunyan. (2014, June). Node.js package for HTTP basic authentication password file utility. [Online]. Available: https://github.com/gevorg/htpasswd/

[56] Axis Communications. (2009, Nov.). Axis 214 PTZ network camera user's manual. [Online]. Available: http://www.axis.com/files/manuals/um_214_37546_en_0911.pdf

[57] E. Marcotte, "Our responsive web," in *Responsive Web Design*, New York: A Book Apart, 2011, pp. 3-4.

[58] J. Keith, "Rich media," in *HTML5 for Web Designers*, New York: A Book Apart, 2010, pp. 22.

[59] The jQuery Foundation. (2014, Apr.). Our project. [Online]. Available: https://jquery.org

[60] Leaflet.js. . (2013, Dec.). Leaflet: an open-source javascript library for mobile-friendly interactive maps. [Online]. Available: http://leafletjs.com

[62] Esri. (2103, Dec.). Esri Leaflet: A lightweight set of tools for using ArcGIS services with leaflet. [Online]. Available: http://esri.github.io/esri-leaflet/

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California